ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN

UNIVERSIDAD POLITÉCNICA DE CARTAGENA

Universidad Politécnica de Cartagena

Proyecto Fin de Carrera

Diseño y desarrollo de un sistema automatizado de control de entradas y salidas de camiones

Titulación: Ingeniero Técnico de Telecomunicaciones
Intensificación: Telemática
Alumno/a: Miguel Alarcón Ortiz
Director/a/s: Alejandro Santos Martínez Sala

Universidad
Politécnica
de Cartagena

Autor	Miguel Alarcón Ortiz
E-mail del Autor	Miguel.Alarcon.Ortiz@gmail.com
Director	Alejandro Santos Martínez Sala
E-mail del Director Codirector (es)	alejandros.martinez@upct.es
Titulo del PFC	Desarrollo de una aplicación para el control de flujo de vehículos.
Descriptores	Control del flujo, vehículos, .NET

Resumen:

El presente proyecto final de carrera pretende estudiar y explorar las posibilidades de las aplicaciones desarrolladas desde la base de la reingeniería de procesos. El objetivo global es el diseño y desarrollo de una aplicación telemática, sobre la plataforma .NET, que sea accesible bajo un computador de mínimas características y que muestre las posibilidades de mejora de la gestión de la información y uso de las TIC. Esta aplicación automatizará y mejorará todos los procedimientos y procesos involucrados en el paso de vehículos a través de una entrada monitorizando toda una plataforma.

El objetivo final es mediante la monitorización de diferentes procesos obtener un control y una seguridad en los datos recogidos por la plataforma para evaluar el potencial en la gestión de la misma.

Titulación	Ingeniero de Telecomunicación, especialidad Telemática.
Departamento	Tecnologías de la Información y las comunicaciones (TIC)
Fecha de Presentación	

Agradecimientos y dedicatoria

Este proyecto se lo dedico a mi familia, a Alejandro Santos y Digital Home s.coop, sin cuya ayuda y apoyo no habría llegado hasta aquí.

Agradecerle al Dr. Alejandro Santos la tutoría de este proyecto, en el que he conseguido aprender a diseñar, evaluar e implementar un sistema automatizado con aplicaciones RFID.

El hecho de desarrollar este proyecto desde el inicio, haciendo el diseño, el análisis, y la implementación me ha ayudado a una mejor comprensión de los conceptos adquiridos en los años de carrera.

A todas las personas que me han ayudado,

Gracias.
Miguel Alarcón

INDICE GENERAL

ÍNDICE DE ILUSTRACIONES.

Capítulo1: Introducción.

1.1 Antecedentes y explicación de la problemática.

Aunque nos encontramos en pleno siglo XXI podemos encontrar multitud de lugares donde las tareas a desempeñar son costosas, artesanales y dan muestras claras de vestigios pasados. Es por ello que mediante las Tecnologías de la Información y las Comunicaciones (TIC) intentaremos en la medida de la posible obtener importantes mejoras en las prestaciones y hacer así más productivas a la par de evitar fallos en el transcurso de las actividades llevadas a cabo.

En los días actuales hemos conseguido desarrollar conocimientos sobre diferentes áreas electricidad, electrónica, telecomunicaciones, etc. Es mediante la unión de algunas de estas áreas por lo que se pueden desarrollar sistemas. Estos sistemas me permitirán reducir la mano de obra, simplificar el trabajo para que así se dé propiedad a algunas máquinas de realizar operaciones de forma automática, es decir, se va a dar un proceso más rápido y eficiente.

El presente proyecto final de carrera pretende estudiar y explorar las posibilidades de las aplicaciones desarrolladas desde la base de la reingeniería de procesos como soporte a una empresa que tiene como base las TIC.

1.2 Herramientas usadas y Formación en distantes Tecnologías empleadas.

En este apartado se detalla cada una de las diferentes herramientas utilizadas para el desarrollo de la aplicación que conforma el proyecto final de carrera, explicando brevemente su funcionamiento y su aporte a la aplicación. En los capítulos tercero y cuarto se apreciará con más detalle la relevancia de las mismas.

1.2.1 Tecnología RFID.

Para la realización de este proyecto final de carrera ha sido necesaria la utilización de tecnología RFID. En concreto se ha usado el **lector** (en inglés, *Reader*) **Symbol XR480**, El principal objetivo de un lector de RFID es transmitir y recibir señales, convirtiendo las ondas de radio de los tags en un formato legible para las computadoras. Esta tecnología ha sido parte fundamental del proyecto pues ha sido la encarga de la labor de identificación.

Los avances en la microelectrónica y la economía de escala han posibilitado el desarrollo de los tags RFID (Radio Frecuency Identification) que son dispositivos

inalámbricos de bajo coste. Un sistema RFID consiste en un lector o estación base que se comunica con los tags cuando éstos pasan por su área de cobertura, leyendo o escribiendo en la memoria de datos que incorporan. Un lector RFID consta de una interfaz radio para comunicarse con los tags, una interfaz para el volcado de datos al sistema de información (Puerto serie, Ethernet, CAN Bus, etc.) y una unidad de control (micro controlador o PC empotrado) que gobierna el dispositivo. La capacidad para crear redes de lectores que se comunican entre sí y con otros sistemas (servidores, bases de datos, autómatas, etc.) permiten una gran versatilidad a la hora de plantearse la automatización de determinados procesos industriales.

1.2.2 Plataforma .NET.

La aplicación que compone este proyecto se ha desarrollado en la plataforma .NET, mediante el entorno de desarrollo Microsoft Visual Studio 2008 Professional, y ha sido codificada en el lenguaje de programación **C Sharp (C#).** Como base de datos se ha utilizado **MySQL**, por ser la que mejor se adaptaba a nuestras necesidades.

El criterio a la hora de elegir tanto la plataforma como la base de datos vino marcado por los anteriores proyectos desarrollados en la empresa, predecesores de éste, y que ya estaban implementados con estas herramientas, pues en el cuerpo técnico estaban familiarizados con estas herramientas de desarrollo.

Todas estas herramientas quedan detalladas más profundamente en el capítulo tercero y cuarto de esta memoria.

1.3 Objetivos del proyecto

El proyecto final de carrera consistirá en el diseño y desarrollo de una aplicación telemática para el control de un sistema automatizado de control de entradas y salidas de camiones. En el que además también en última instancia se automatizara el proceso de facturación en una planta de reciclaje.

Se plantean los siguientes objetivos parciales:

- Se van a estudiar y definir procesos operativos sobre la realización de las acciones que ha de llevar a cabo un camión para definir un correcto funcionamiento del circuito a seguir.

- Se van a proponer y evaluar una serie de funcionalidades de valor añadido que nos proporcionarán una mayor calidad del control del sistema, como son los logs o la entrega de tickets.

- Se va a diseñar, desarrollar y poner en marcha dicha aplicación mediante la programación de simuladores que permitan el desarrollo del software sin necesidad de contar con los elementos físicos del sistema, que serán integrados una vez desarrollada la aplicación.

El objetivo global es el diseño y desarrollo de una aplicación telemática, sobre la plataforma .NET, desarrollado más concretamente en C# y que muestre las posibilidades de mejora de la gestión de la información y uso de las TIC. Esta aplicación automatizará y mejorará todos los procedimientos y procesos involucrados en el desarrollo de las acciones del camión paso a paso y en la detección de errores que puedan ocurrir en el sistema. El objetivo final es automatizar un proceso que en la actualidad goza de escasa tecnología y así la mano de obra sólo quedaría encargada de la supervisión del sistema, mediante la aplicación.

1.4 Estructura del resto de la memoria.

El resto de la memoria del presente proyecto se estructura en los siguientes capítulos:

El capítulo segundo consiste en un análisis, diagnóstico y primera toma de contacto con el funcionamiento de un sistema que maneja un gran flujo de vehículos entrando y saliendo en este capítulo haremos un análisis previo del funcionamiento del sistema y los estados del mismo.

En el capítulo tercero se expondrá de manera más ampliada la arquitectura del sistema que se ha adoptado para la solución así como los elementos que la componen y la relación entre los mismos. Se va a ver la imagen global del sistema para posteriormente centrarse en elementos concretos y explicar su funcionamiento.

En el capítulo cuarto estará compuesto de la solución software que hemos desarrollo para la solución de la problemática explicada en el capítulo dos. Diseño de tablas de la base de datos, a partir de ahora ***BBDD***, implementación de los distintos hardware utilizados y la capa lógica encargado de la principal funcionalidad de la aplicación.

En el capítulo quinto se realizara un pequeño análisis de la financiación de la ejecución del proyecto y la rentabilidad que proporciona en comparación con el sistema de uso actual de la planta de reciclaje.

En el capítulo sexto para finalizar explicaremos hasta donde se ha llegado y las posibles líneas futuras que pueden interesar para continuar el desarrollo de la aplicación.

Capítulo 2: ANÁLISIS DEL PROCESO PRODUCTIVO Y REINGENIERÍA DE LOS PROCESOS

2.1 Introducción

El presente proyecto final de carrera consiste en una serie de mejoras del actual modelo de trabajo que siguen algunas plantas de reciclaje que gestionan la descarga de residuos en sus terrenos. Este presente proyecto final de carrera se ha desarrollado por el alumno Miguel Alarcón Ortiz mediante la realización de prácticas de empresa. En la empresa Digital-Home S. Coop. Para el desarrollo en su totalidad del proyecto el alumno ha invertido un total de 420 horas en la labor técnica. En la labor técnica el alumno ha desarrollado una maqueta para un caso genérico de gestión de entradas y salidas de camiones, tratando así de simular un escenario real.

Este proyecto tiene por objeto agilizar procesos mediante avances tecnológicos, no utilizados hasta la fecha, que iremos incorporando según las necesidades que vayamos detectando en la etapa de diseño.

El presente proyecto final de carrera surge por la necesidad de la empresa de reducir el tiempo total de los procesos para la realización de las acciones propias de los vehículos, con el consiguiente recorte de los costes, y de la mano de obra debido a la evolución que han sufrido los sistemas de trabajo. Este proyecto se abordó desde la perspectiva de la reingeniería de procesos. Se estudió el flujo clásico de procesos de ingeniería-producción para la realización de una carga o descarga de material por un vehículo en la instalación, habiéndose hecho hincapié en aquellas partes susceptibles de ser rediseñadas y adaptándolas a las nuevas tecnologías.

2.2 Descripción procesos actuales

En este apartado se evaluará cómo se comporta el sistema actual y se hará una clasificación de los procedimientos, que son las reglas a seguir, que ha de realizar el vehículo para que tenga éxito una acción en el sistema.

2.2.1 Procedimiento de entrada

El procedimiento actual de entrada es actualmente un procedimiento que se realiza prácticamente en su totalidad de forma manual y con un elevado porcentaje de intervención humana. Se procederá a narrar el esquema de trabajo empleado actualmente en una planta de reciclaje.

Actualmente en el sistema intervienen distintos tipos de vehículos, cuyas características y restricciones técnicas se han tenido en cuenta a la hora de especificar una solución propuesta.

El camión que desea entrar al sistema se encuentra con una serie de impedimentosSe carece de señales que indiquen las maniobras a realizar o el estado en el que se encuentra la planta. Por parte del camión, para sortear dichas trabas es casi siempre necesario que esté presente un operario en la cabina a pocos metros de la entrada para ir guiando a los conductores de los camiones a realizar dichas maniobras, como apertura de barreras, con éxito y así dar paso al camión. El procedimiento de entrada resulta sencillo.

2.2.2 Procedimiento de salida

Este procedimiento es exactamente igual que procedimiento de entrada a excepción de varios puntos, pues como se había mencionado antes, la intervención del operario de cabina es necesaria para el transcurso correcto de los vehículos. La única diferencia resaltable con el procedimiento de entrada es que el operario de cabina da por supuesto que si el vehículo sale de la planta éste habrá sido identificado en el procedimiento de acceso a la planta, de modo que no vuelve a comprobar la identificación del vehículo. En este punto resulta necesaria la intervención de dicho operario para generar un ticket/recibo que contendrá algunos parámetros como el tipo de residuo que transporta el camión, la cantidad de peso que ha cargado/descargado, la matrícula, el conductor, etc. Esta generación de ticket se hace de forma manual por parte del operario metiendo todos los parámetros en una máquina, ver imagen en figura 2.1, para su posterior impresión (ver imagen en figura 2.2). Esta forma de llevar a cabo la facturación no contempla la posibilidad de un registro fácilmente accesible, porque si se desea comprobar una factura habría que rebuscar entre todas las facturas generadas, ya que no se guarda ningún control a la hora de clasificar los ticket/recibos.

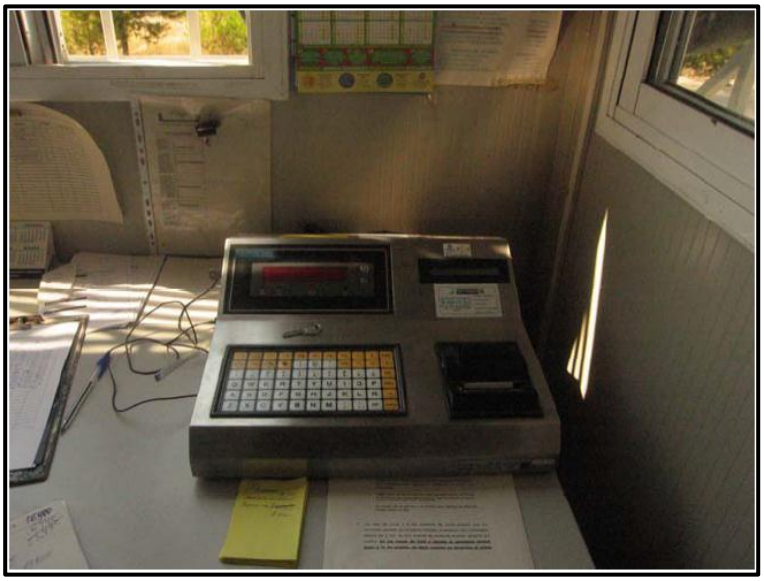

Figura 2.1: Máquina de facturación manual.

Figura 2.2: Máquina de recogida de ticket.

2.2.3 Procedimiento de pesada

En el procedimiento de pesada intervienen varias máquinas aunque aún así sigue presentando un alto nivel de intervención humana porque todas las comprobaciones las ha de supervisar el operario. Así por ejemplo, la comprobación del tipo del residuo que transporta el camión requiere que el operario se acerque a la parte de carga del vehículo a examinarlo). También se ha de comprobar la correcta situación en la báscula, que la pesada sea acorde con el producto que lleva, es decir, acotar la carga que está cargada en el vehículo. Para lanzar el cálculo de la carga el operario ha de volver al interior de la cabina y pulsar la tecla que activará el peso industrial.

Figura 2.3: Vehículo entrando en la plataforma.

2.2.4 Procedimiento de identificación

Para la realización de este procedimiento, es de nuevo necesaria la intervención humana para hacer corresponder la matrícula del vehículo con el camionero en el registro que tiene el sistema que en muchos casos resulta algo tan sencillo como una tablilla, ver figura 2.4, en los que irá apuntando manualmente los datos propios hora a la que entro el vehículo, conductor del vehículo, hora del evento, etc.

Es fácil comprobar que mediante este método queda patente la gran probabilidad de fallo y posterior desajuste en las entradas y salidas de la planta que están registradas en la tablilla, pues al ser un procedimiento tan repetitivo el operario poco a poco tenderá a no prestarle toda la atención necesaria o simplemente no se hace la letra lo suficientemente legible.

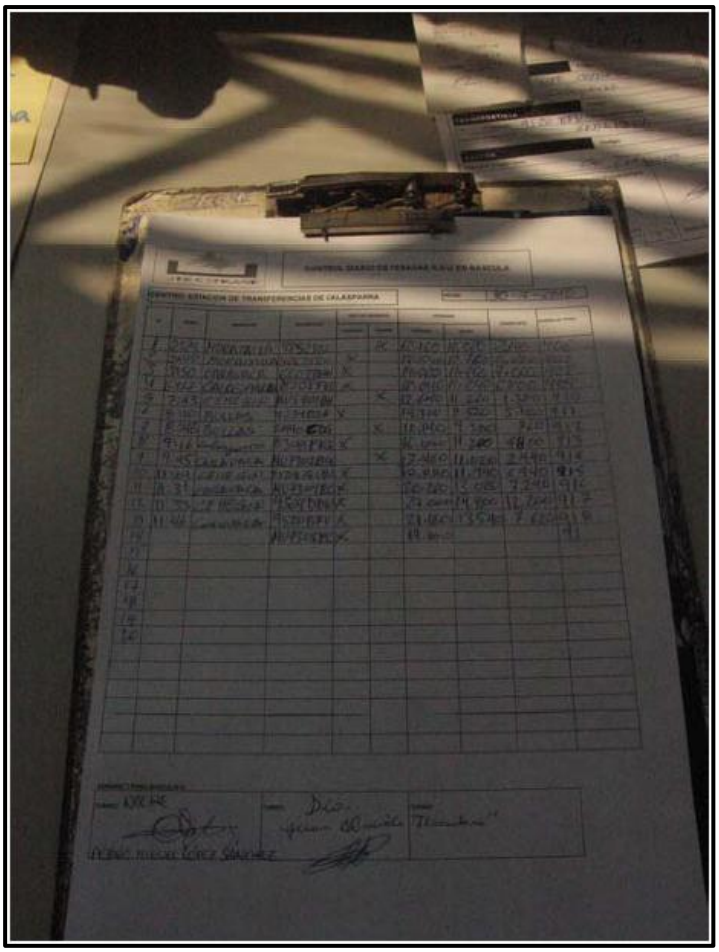

Figura 2.4: Tabla que sirve de registro.

2.3 Reingeniería de los procesos

La reingeniería de procesos es el rediseño radical de los procesos para lograr mejoras dramáticas en medidas de desempeño tales como en costos, calidad, servicio y rapidez, busca llegar a la raíz de las cosas, no se trata solamente de mejorar los procesos, sino y principalmente, busca reinventarlos.

Figura 2.5: Escala de reingeniería de procesos

2.3.1 Descripción de los procesos basados en la reingeniería

Vista la solución actual que se contempla para solventar el problema de la planta de reciclaje y basándose en la experiencia desarrollada por la metodología de reingeniería de los procesos vamos a proporcionar una nueva solución, a nuestro criterio la más correcta, redefiniendo de nuevo los procesos a seguir para la planta.

En este apartado, se realizará el estudio del sistema como se contempla por el equipo técnico, vamos a definir el comportamiento global para encontrar la solución deseada se describirá la situación para entender los requisitos necesarios y cómo llevarlos a cabo. Sería interesante hacernos una pregunta a la que contestar.

¿Cómo queremos que sean los procesos para mejorar/automatizar?

El camión se aproxima al sistema y visualiza el semáforo del cual podrá observar si tiene acceso o si se ha de esperar. Este semáforo se encontrará en verde por defecto a excepción del momento en que se encuentre un camión dentro de los procedimientos Entrada/Salida .Se plantean dos casuísticas:

- Si éste se encuentra en rojo el camión deberá esperar hasta que este pase a ser verde
- Si éste se encuentra en verde el camión se aproximará a la barrera de entrada.

En este momento:

Se activará el sensor de posición o fotocélula que detectará que el camión intenta entrar en la báscula.

Se activará el sistema de reconocimiento de matrículas. El cual reconoce el patrón de la matrícula en la imagen tomada y compara el resultado obtenido con la base de datos existente. Se activará la identificación por RFID (Identificación por radiofrecuencia).En este intervienen tres elementos: antena RFID (fija en el punto de control), tag pasivo (instalado en el camión) y lector fijo de RFID (Reader).De forma análoga el sistema de reconocimiento de matrículas se procederá a comparar el "tag/identificador" con la base de datos de vehículos autorizados existente.

Este doble sistema de reconocimiento se proyecta para conseguir un sistema redundante que sea completamente tolerante a fallos. El diseño propuesto permite que en el caso de que uno de los sistemas de identificación falle el otro sistema actúe como "*backup*", aumentando la fiabilidad del sistema. La intervención humana pasa a encargarse de la resolución de incidencias (que se notificaran /registraran adecuadamente a través del software de control).

Si uno de los dos sistemas o ambos reconoce al vehículo como autorizado la barrera de entrada a la báscula se abrirá, simultáneamente se iluminará un piloto verde que confirmará que el proceso se ha realizado correctamente. En el caso contrario se iluminará un piloto naranja, tras el cual el conductor deberá ponerse en contacto con el operario.

El camión entra dentro de la báscula y avanza hasta la posición de pesada. Para controlar que el camión se encuentra en la posición adecuada se utilizará un nuevo sensor detección, éste se activará cuando la puerta del conductor se encuentre a la altura de la peana de selección de residuo/residuos.

Una vez seleccionado el tipo de residuo se iniciará la pesada (la báscula se comunica con el servidor /software de gestión a través de una comunicación por puerto serie ya existente). Este valor se procesará, si este se encuentra dentro de los parámetros esperados:

- El display mostrará un mensaje con la pesada, la matrícula, la hora de entrada y el tipo de residuo.
- se tomará una instantánea de control .
- se iluminará un piloto verde que indica que esta se ha realizado correctamente.
- se abrirá la barrera de salida.

En el caso de que los valores de pesada no se encuentren dentro de los márgenes esperados, lo más probable es que el conductor haya cometido un error en la selección del tipo de residuo; en el display se mostrará un mensaje requiriendo que vuelva a realizar esta nueva con un número máximo de 3 intentos, si fallase ese número de intentos se procedería a llamar al operario, la selección y se realizará una nueva pesada.

Una vez que el camión sale de la báscula, se cerrará la barrera y los semáforos pasarán a su estado por defecto ambos ámbar.

El camión que quiere salir visualiza el estado del semáforo .Este semáforo (pilotos rojo, ámbar) se encontrara en ámbar por defecto, a excepción del momento en que se encuentre un camión dentro del procedimiento de entrada/salida. Se plantean dos casuísticas:

-Si este se encuentra en Rojo el camión deberá esperar hasta que éste pase a ser ámbar

-Si éste se encuentra en ámbar el camión se aproximara a la barrera de entrada .En este momento:

Se activará el sensor de posición o fotocélula que detectara que un camión intenta entrar en la báscula. Se procederá a la apertura de la barrera.

El camión entra dentro de la báscula y avanza hasta la posición de ticket. Para controlar que el camión se encuentra en la posición adecuada se utilizará un nuevo sensor de detección, éste se activara cuando la puerta del conductor se encuentre a la altura de la peana de solicitud de ticket.

Una vez solicitado el ticket se iniciará una nueva pesada (la báscula que se comunica con el servidor/software de gestión a través de una comunicación por puerto serie ya existente) y se envían estos datos al servidor que procesará esta información y la ya almacenada para la pesada de entrada.

A continuación:
- Se procederá a imprimir el ticket (el conducto de pasar a recogerlo por la cabina). Con los siguientes datos:

- Hora de entrada /hora de salida
- Tipo de camión
- Valor de la pesada
- Matrícula
- Tag

- Se tomará una instantánea de control

Se abrirá la barrera de salida.
Una vez que el camión sale de la báscula, se cerrará la barrera y los semáforos pasarán a su estado por defecto (ambos en ámbar).

2.4 Deficiencias y puntos de mejora de los procesos

En este apartado se pretende demostrar mediante una tabla las deficiencias que había hasta el momento y las soluciones tomadas.

Procedimientos	**Antiguos**	**Nuevos**
Entrada	✓ Carece de señales en el sistema. ✓ Es necesario que esté presente un operario de planta. ✓ Identificación manual.	✓ Semáforo indicando el estado del sistema. ✓ No es necesaria la presencia de un operario de planta. ✓ Identificación automática y redundante.
Salida	✓ Carece de señales en el sistema. ✓ Es necesario que esté presente un operario de planta. ✓ Los datos se toman de forma manual. ✓ Mala gestión de los log.	✓ Semáforo indicando el estado del sistema. ✓ No es necesario la presencia de un operario de planta. ✓ Se comprueba la identificación. ✓ Los datos de toman de forma automática. ✓ Control de gestión de los log.
Pesada	✓ Es necesario que esté presente un operario de planta. ✓ Comprobación manual de la correcta situación del vehículo. ✓ Pesada con intervención del operario de planta.	✓ No es necesaria la presencia de un operario de planta. ✓ Automatización de la correcta posición del vehículo. ✓ Pesada automática.
Identificación	✓ Es necesario que esté presente un operario de planta. ✓ Hay un leve control del registro.	✓ No es necesaria la presencia de un operario de planta. ✓ Mediante tecnología RFID y comprobación de matrícula en base de datos.

Como se puede comprobar, gracias al presente proyecto la función del operario de planta pasa a ser de mera supervisión, dado que con los nuevos procedimientos no es necesaria su presencia en el sistema siempre y cuando no se produzcan incidencias.

Algunos puntos en los que resultaría interesante avanzar pueden ser:

Un punto interesante para continuar con el desarrollo sería el siguiente: en la hipotética situación de dos vehículos que se encontrasen a la entrada/salida, uno pendiente de efectuar el procedimiento de entrada y otro el de salida, dar prioridad al que desee

ejecutar el procedimiento de salida de manera que se evitara congestión dentro de la planta.

Tal vez otro punto pudiera ser el de funcionar con varias réplicas del sistema a la vez, es decir, tener el control de varios sistemas para dotar a la planta de más entradas y salidas.

Podríamos añadirle funcionalidades que debido al alcance del proyecto han quedado fuera, tales como el control de un display o una cámara que controle el tipo de residuo que lleva el vehículo.

También resultaría conveniente que se establecieran unos códigos de error para cuando se producen fallos en el sistema poder identificar rápidamente el problema.

Como última mejora se propondría cambiar la tarjeta adquisición de datos, pues actualmente la comunicación con la misma se hace a través del puerto usb y éste no permite obtener la suficiente distancia, es decir, el pc en el que corre la aplicación tendría que estar pegado al sistema.

Capítulo 3: Arquitectura física de la solución.

3.1 Introducción.

Para llegar a comprender en su plenitud la arquitectura e implementación de la solución desarrollada en este proyecto, primero se va a presentar la visión global de la arquitectura propuesta esquematizada en módulos para luego explicar cada uno de los elementos que los componen. Terminando por describir los objetos que las componen y su modo de utilización.

3.2 Diagrama de bloques arquitectura.

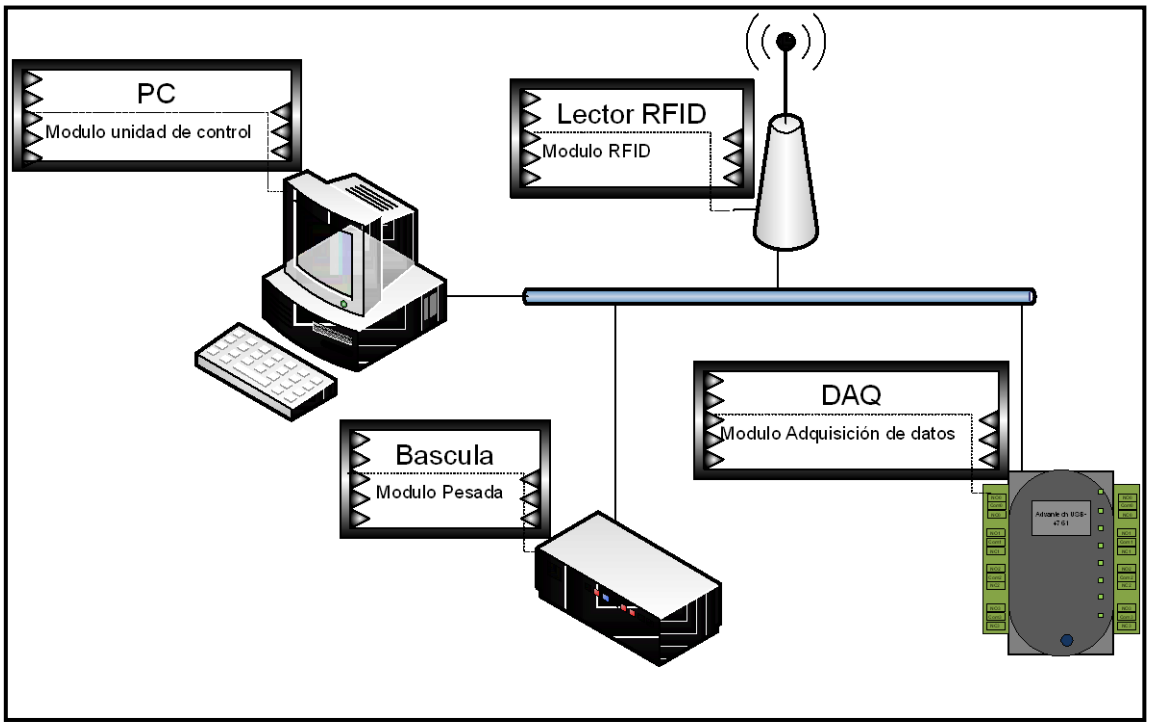

Figura3.1: Arquitectura física del sistema

3.2.1 Modulo adquisición datos

Consiste en la toma de muestras del mundo real (sistema analógico) para generar datos que puedan ser manipulados por un ordenador u otras herramientas electrónicas (sistema digital). Consiste, en tomar un conjunto de señales físicas, convertirlas en tensiones eléctricas y digitalizarlas de manera que se puedan procesar en un Pc. Se requiere una etapa de acondicionamiento, que adecua la señal a niveles compatibles con el elemento que hace la transformación a señal digital. El elemento que hace dicha transformación es el módulo de digitalización o tarjeta de Adquisición de Datos (**DAQ**). En nuestro caso hemos utilizado la tarjeta, BioDAQ-USB 4761 que consta de 8 entradas digitales y 8 salidas digitales, para el control del sistema. A esta tarjeta hemos adaptado una serie de elementos tales como semáforo, barrera, sensores que debemos separar en elementos de entrada y elementos de salida.

Elementos de Salida : como elementos de salidas digitales de la tarjeta DAQ podemos encontrar en nuestro sistema barreras y semáforos ,estos elementos me va condicionar el movimiento dentro del sistema , es decir, en función del estado de la lógica de control que me encuentre tomaran unos estados y otros.

Estos elementos se han simulado mediante agrupación de led.

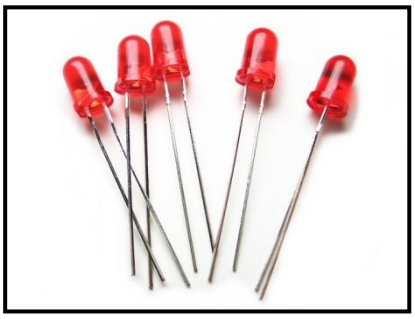

Figura 3.2: Led.

Elementos de Entrada: como elementos de entradas digitales de de la tarjeta DAQ podemos encontrar final de carrera y pulsadores que me permitirán obtener información de la arquitectura del sistema para poder cambiar de unos estados a otros.

Figura 3.3: Final de carrera

Figura 3.4: Pulsador

3.2.2 Modulo unidad de control:

La unidad de control el elemento que se encarga de sincronizar las acciones que realiza cada una de las unidades funcionales de mi arquitectura; en nuestro caso, será un PC. Las funciones de la unidad de control son básicamente dos.

Interpretación de las instrucciones: La unidad de control debe ser capaz de decodificar los códigos de operación que le llegan de mi tarjeta DAQ y actuar de forma diferente para cada uno de ellos.

Secuencia de las operaciones: va a basar su comportamiento en una máquina de estados, en la que se van a seguir unos estados a otros dependiendo del estado de mis entradas y salidas de la tarjeta DAQ.

Figura3.5: PC atom

3.2.3 Modulo RFID.

Módulo RFID que constara de 2 componentes un lector RFID y Tags. Este módulo me proporcionara una identificación a los objetos móviles de mi arquitectura, en nuestro caso usaremos RFID Pasivo y se trabajará en UHF, que es donde trabajan estos sistemas. Como hemos apuntado antes vamos a usar RFID pasiva y sus principales limitaciones son un área de cobertura limitada, que puede oscilar entre 1 cm y los 2 metros, (dependiendo de la longitud de la antena y de la potencia de transmisión). Debido a la naturaleza de nuestra arquitectura resulta innecesaria la escritura de los tags pues este módulo me proporcionará la identificación en mi sistema y nos bastará con la lectura.

Lector RFID: el lector utilizado en este proyecto final de carrera es un Symbol XR480 cuya finalidad está orientada a comunicarse con la antena y la unidad de control, a la que brindará la información obtenida de la antena como el identificador del tag leído, clase a la que pertenece el tag, la antena que ha ofrecido la lectura, etc.

Figura 3.6 Lector RFID

Tag: no es más que una antena de reducidas dimensiones capaz de emitir y recibir. El utilizado para la implantación de este proyecto ha sido un tag pasivo.

Algunas de sus características son:

- Un espacio de memoria pequeño (memorias tipo EEPROM).
- La principal ventaja es que se pueden obtener tags de muy reducidas dimensiones y a un bajo precio.

Figura 3.7: Tag pasivo

Antena: tiene el objetivo de emitir y recibir ondas electromagnéticas, proporcionadas por el lector RFID y captadas por los tags. Las antenas dependen de la relación entre sus dimensiones y la longitud de onda de la señal de radiofrecuencia transmitida o recibida, por eso para el desarrollo de esta tarea se va a usar una antena AN400 de Motorola.

Figura 3.8: Antena

3.2.4 Módulo Pesada.

El módulo de pesada estará compuesto por una báscula conectada por Ethernet cuya única función será la de pesar el vehículo y dar los resultados oportunos a la unidad de control.

Figura 3.9: Báscula

3.3 Diseño y montaje de maqueta.

Introducción.

Para obtener una correcta visión de la problemática y poder adelantarnos a los fallos técnicos y corregirlos para no errar en el proceso productivo, se ha llevado a cabo el diseño y montaje de una maqueta para la realización de pruebas, prevención de fallos y aproximaciones a la solución más óptima.

Figura 3.10: Maqueta del proyecto terminada.

3.3.1Esquema eléctrico de las entradas a la tarjeta I/O.

A continuación os dejo el conexionado de las entradas de la tarjeta a la maqueta.

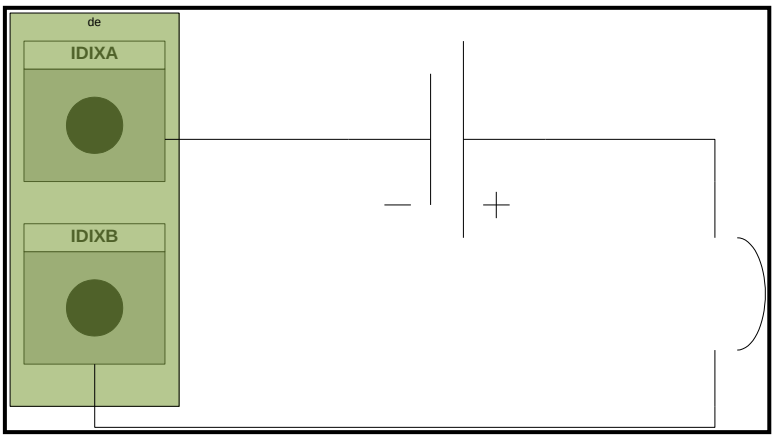

Figura 3.11: Esquema entradas DAQ.

Las cajas de la izquierda se corresponden con las bornas de la tarjeta DAQ mientras que los sensores se encuentran dibujados como un interruptor a la derecha.

3.3.2 Esquema eléctrico de las salidas a la tarjeta I/O.

A continuación os dejo el conexionado de las salidas de la tarjeta a la maqueta.

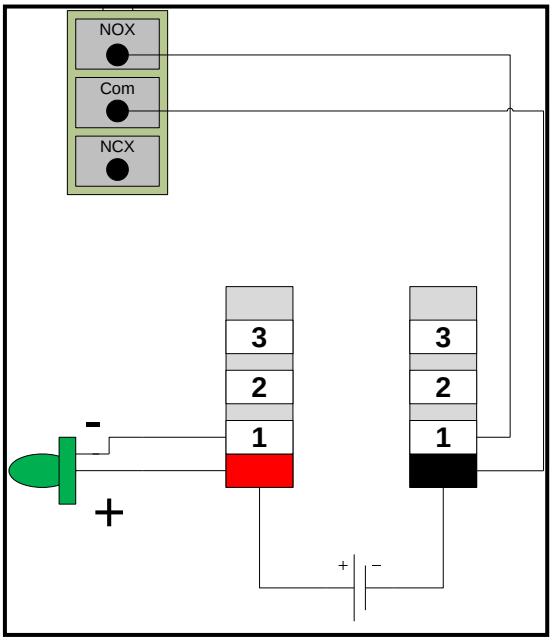

Figura 3.12: Esquema salidas DAQ.

Como en el esquema de las entradas anterior, arriba se encuentran las bornas de la DAQ, el led abajo a la izquierda y el circuito del medio me permite simular en mi maqueta el sistema eléctrico real.

3.3.3Descripción eléctrica del circuito completo.

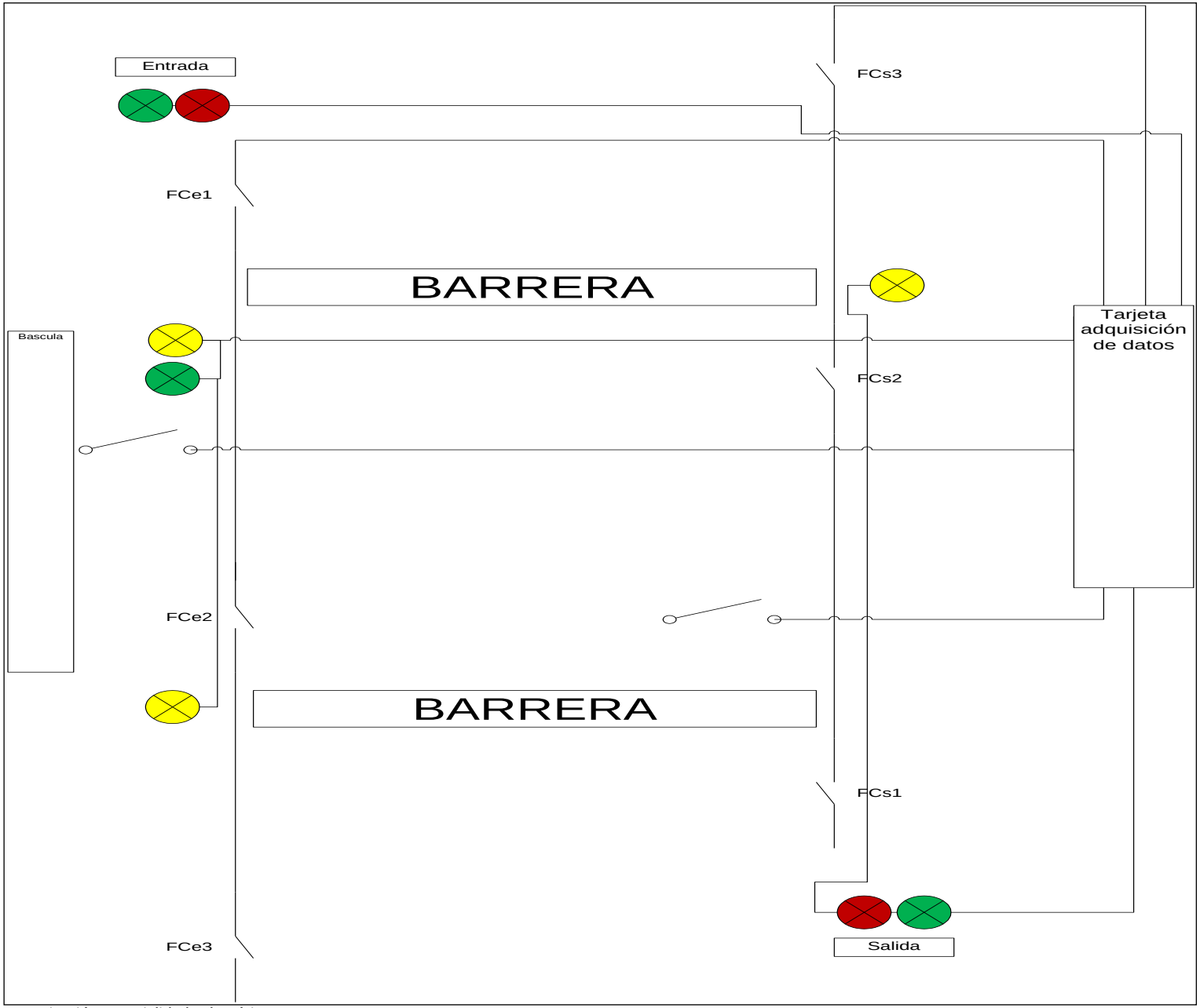

Figura 3.13: Esquema Eléctrico General

Capítulo 4: Arquitectura Software de la solución

4.1 Introducción.

Al igual que se hizo en el capítulo anterior se proporcionará una visión general de la arquitectura software para poco a poco poder ir conociendo los detalles de implementación del sistema. En la aplicación telemática se pueden diferenciar claramente las siguientes funcionalidades.

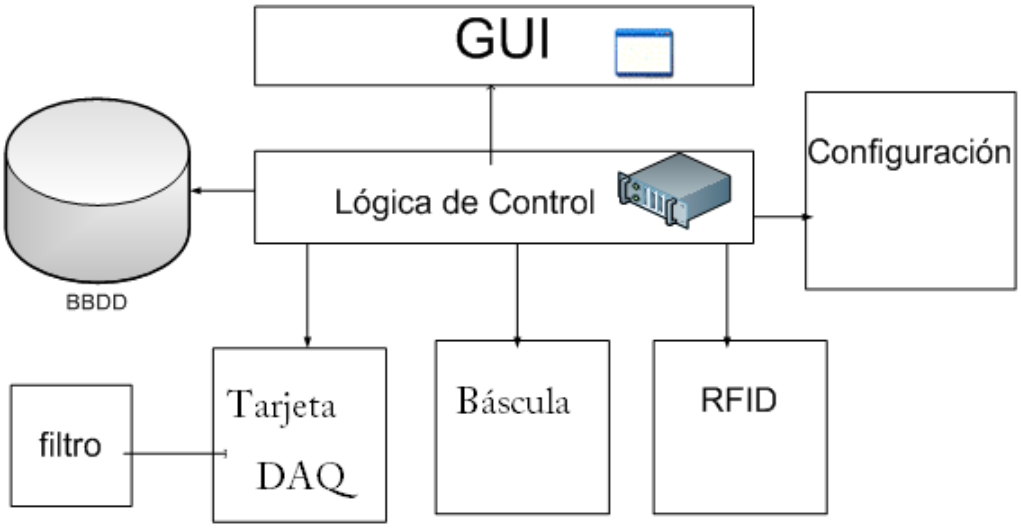

Figura 4.1: Arquitectura software del sistema.

- **GUI:** Esta parte de la aplicación se corresponderá con la interfaz gráfica de la misma y no tiene ninguna funcionalidad al respecto.
- **BBDD:** Es la base de datos de la aplicación, en la que la aplicación almacenará toda la información generada durante el proceso.
- **Lógica de control:** En esta parte de la aplicación es donde se llevará a cabo el control de los elementos hardware y software del sistema necesarios para realizar cada uno de los procedimientos.
- **Configuración:** Se implementará un sistema de log, para comprobar el correcto funcionamiento del sistema así como registrar cualquier incidencia que ocurra durante la ejecución de los distintos procedimientos.
- **Filtro:** Implementado por la necesidad de revolver un problema físico de la arquitectura facilitará la obtención de datos de la tarjeta DAQ.
- **Tarjeta DAQ:** Implementación software para el uso del dispositivo DAQ
- **Báscula:** Implementación software para el uso del dispositivo Báscula.
- **RFID:** Implementación software para el uso del dispositivo RFID.

4.2 Diagrama de bloques de la arquitectura software.

4.2.1 Diagrama de clases.

Un requisito que se ha tenido en cuenta a la hora de realizar la programación es que las partes hardware sean intercambiables, es decir, que los distintos dispositivos puedan ser sustituidos por otros, por ejemplo usar un modelo diferente de tarjetas de adquisición de datos o de lector RFID, sin tener que modificar la estructura principal del programa. La solución por la que se ha apostado es la de utilizar el concepto de polimorfismo, mediante la implementación de una clase abstracta para cada tipo de dispositivo de modo que cuando haya que sustituir algún dispositivo se realicen los cambios mínimos estrictamente necesarios en el código de la aplicación. Esto es, crear una nueva clase que herede de la clase abstracta del tipo de dispositivo que corresponda e implemente sus métodos abstractos haciendo las veces de middleware del nuevo dispositivo. Esta solución ha permitido desarrollar fácilmente los emuladores con las diferentes funcionalidades, como tarjeta DAQ, captación de imágenes y el sistema RFID, que permiten indicar a la aplicación si queremos usar los dispositivos reales o emular cualquiera de ellos. Cabe destacar que se optó por utilizar clases abstractas en lugar de interfaces porque existían métodos comunes a cualquier dispositivo que con el uso de clases abstractas pueden ser implementados y heredados directamente por las clases hijas, mientras que de haber usado interfaces tendrían que haber sido reescritos cada vez que se creara una nueva clase que debiera implementar alguna de dichas interfaces.

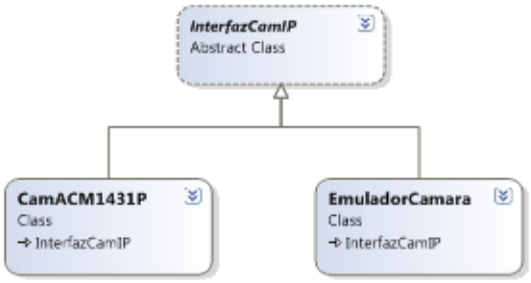

Figura 4.2: Diagrama de clases del dispositivo de captación de imagenes

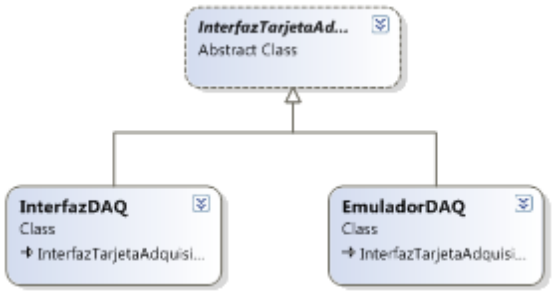

Figura 4.3: Diagrama de clases DAQ

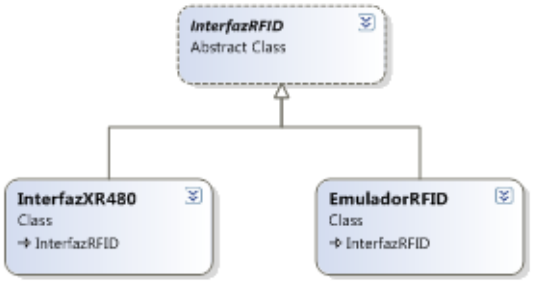

Figura 4.4: Diagrama de clases RFID

A la hora de implementar la interfaz gráfica se ha utilizado un esquema de trabajo similar, creando una clase abstracta llamada *SignalListenerGraphicComponent* de la que hereda cualquier componente que deba ser pintado en la interfaz. Esta clase contiene un método abstracto para pintar, que han de implementar todos los objetos que hereden *SignalListenerGraphicComponent*.

Las clases que heredaran de *SignalListenerGraphicComponent* son *Barrera*, *PlantaSistema*, *Semaforo* y *Vehiculo*. Estas clases además de implementar el método abstracto presentan dos propiedades adicionales: posición y estado. Con la propiedad *posición* podremos situar el objeto en un plano de 2 dimensiones y con la propiedad *estado* podemos especificar el estado en el que se encuentra el objeto. Así, por ejemplo, en el caso de un semáforo éste puede estar en ámbar o en rojo, estando indicado en la propiedad *estado*. Estas propiedades son comunes para todos los objetos excepto para PlantaSistema pues para este objeto se omite la propiedad de estado ya que siempre se va a encontrar en el mismo estado y no es necesario que se realice ningún cambio.

La clase *AreaGrafica* es la que servirá de lienzo para dibujar todos los componentes gráficos. Esta clase hereda del componente *PictureBox*, esto se hace así para heredar el método *OnPaint* de la clase *PictureBox* y poder dibujar los gráficos. Al implementar todos los objetos el método de la clase abstracta de la que heredan bridan la posibilidad de utilizar el método *OnPaint* de una forma muy sencilla.

El principio de funcionamiento es simple: cuando la aplicación principal invoque el método pintar de un objeto que hereda de *SignalListenerGraphicComponent*, éste se encargará de pintarse a sí mismo en el área gráfica en función de su estado, de modo que añadir nuevos objetos al sistema sería muy simple y "limpio", puesto que la aplicación principal no necesita saber de qué clase concreta es cada objeto a pintar, le basta con saber que dispone del método pintar (porque hereda de la clase *SignalListenerGraphicComponent)*.

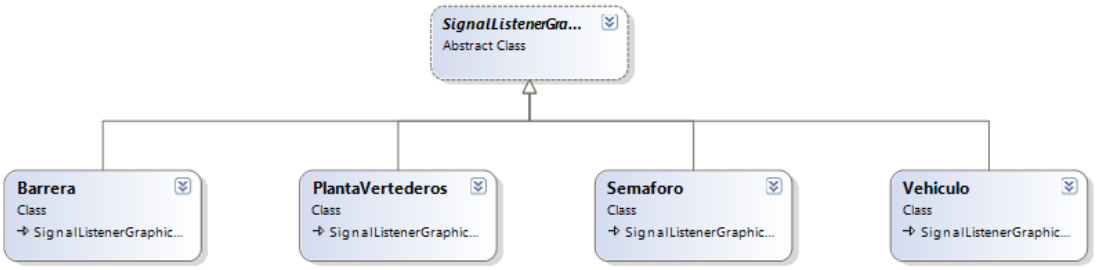

Figura 4.5: Diagrama de clases de los gráficos.

Adicionalmente, cabe destacar que se ha utilizado el patrón *Observador* para la captación de los eventos que puede generar cada uno de los elementos del sistema (el evento que lanza el controlador del dispositivo de capturar una imagen, los que genera la tarjeta de adquisición cuando hay un cambio en las entradas, la lectura de un tag por parte del lector RFID, etc.), de modo que la clase encargada de la lógica de control se subscribe como observador del controlador de cada dispositivo de modo que éste le notifique cada vez que ocurra un evento.

4.2.2Descripción de las clases.

A continuación se comentarán las funcionalidades de todas las clases empleadas en este proyecto. Se hará una división en bloques correspondiente a cada una de las partes desarrolladas.

Comenzaremos con clases abstractas que deben ser heredadas por todos los dispositivos:

InterfazCamIP: Clase abstracta que debe heredar cualquier el controlador de cualquier dispositivo encargado de la captura de imagenes que se pretenda utilizar**.**

InterfazReconocimentoMatricula: No era necesario desarrollarla pues es muy difícil que se emplease otras librerías para el reconocimiento de gráficos pero se decidió para mantener la filosofía que se había llevado a cabo con los distintos dispositivos.

InterfazRFID: Clase abstracta que debe heredar el controlador de cualquier lector RFID que se pretenda utilizar. Heredan de esta clase el controlador del reader XR480 y el controlador del simulador RFID.

InterfazTarjetaAdquisicionDatos: Clase abstracta que debe heredar el controlador de cualquier tarjeta o dispositivo de adquisición que vaya a controlar los sensores y las luces del sistema.

A continuación se listan los controladores de los dispositivos hardware específicos que se han utilizado en el proyecto.

ControladorCAMACM1431P: Clase que implementa el controlador de un dispositivo para capturar imágenes que se utiliza en el presente proyecto (parcialmente implementado). Hereda de la clase abstracta ***InterfazCamIP***.

ControladorBBDD: Clase que gestiona la comunicación con la base de datos y permite realizar consultas.

ControladorTarjetaAdquisicionDAQ: Clase que implementa el controlador del modelo de tarjeta de adquisición de datos utilizada en el proyecto. *Hereda de la clase abstracta* ***InterfazTarjetaAdquisicionDatos***.

ControladorReaderXR480: Clase que implementa la capa middleware para controlar el modelo de lector RFID utilizado en el proyecto. Hereda de la clase abstracta *InterfazRFID.*.

ReconocMatricula_Mil: Clase no creada en el proyecto pero implementada para trabajar en el reconocimiento de caracteres.

Clases que no guardan ninguna relación entre ellas, solo nos aportan funcionalidad.

ClsReader: Clase que permite controlar el lector RFID XR 480 y nos es proporcionada por *Motorola.*

ConfigReader: Clase utilizada para establecer la configuración apropiada del lector, se encarga de crear la configuración con los parámetros necesarios.

DatosDescarga: Clase cuya única función es la de encapsular los datos obtenidos tanto a la entrada como a la salida de un único vehículo**.**

EscribeFichero: Clase que encapsula la escritura en ficheros. Resulta muy útil a la hora de mantener el sistema de log's.

Estado: Clase que proporcionará una encapsulación de los posibles estados en los que se puede encontrar el camión.

FiltroTarjetaAdquisicion: Clase utilizada para filtrar eventos que genera la tarjeta de adquisición de datos, ignorando los impulsos eléctricos que no sean necesarios. Mantiene una estrecha relación con la clase *Reglas*.

GenerarPDF: Clase que sirve para generar PDF's mediante una librería externa de uso libre, usada para generar los tickets.

Form1: Clase principal de la aplicación telemática.

Reglas: Clase utilizada para definir qué condiciones hay que comprobar cuando ocurra un evento estando en un estado y, en caso de cumplirse, a qué estado hay que pasar y qué acciones hay que realizar**.**

Para el desarrollo de la aplicación han sido desarrollados distintos emuladores (del lectorRFID, dispositivo de captación de imágenes y de la tarjeta de adquisición de datos) que proporcionan un escenario óptimo para continuar el desarrollo y depuración de la aplicación (especialmente de la lógica de control) sin necesidad de disponer del montaje hardware.

EmuladorCamara: Clase que emulará un dispositivo de captación de imágenes reales, ofreciendo la misma API de control y lanzando los mismos eventos que generaría una dispositivo real si estuviese conectada a la aplicación. Devuelve una imagen preseleccionada por el usuario desde la clase *FormEmuladorCamara* cuando la clase principal (lógica de control) le solicite capturar una imagen.

EmuladorDAQ: Clase que emulará una tarjeta de adquisición de datos real con 8 entradas y 8 salidas, ofreciendo la misma API de control y lanzando los mismos eventos que generaría una tarjeta real. Ofrece una interfaz gráfica donde se puede apreciar el estado de las entradas y salidas.

EmuladorRFID: Clase que emulará un lector RFID real, ofreciendo la misma API de control y lanzando los mismos eventos que generaría uno real si estuviese conectado a la aplicación. Su uso es muy simple, se limita a recoger el identificador de tag especificado por el usuario en la clase *FormEmuladorRFID* y lo pasa a la clase principal.

FormEmuladorCamara: Interfaz gráfica que me simula un dispositivo de captación de captación de imágenes y permite cambiar dichas imágenes a voluntad**.**

FormEmuladorDAQ: Interfaz gráfica que me permite simular una tarjeta adquisición de datos observando el cambio de color en las salidas y cambiando el estado de los checkbox en las entradas.

FormEmuladorRFID: Sencilla interfaz gráfica que me simula la lectura de un tag RFID proporcionado un recuadro donde indicar diversos parámetros del tag**.**

En esta parte se dará una breve explicación sobre las clases que componen el área gráfica y para que se han utilizado.

SignalListenerGraphicComponent: Clase de la que van a heredar todos los componentes gráficos y que permitirá tener un mayor control del proceso de pintado de componentes en el área gráfica.

AreaGrafica: Clase que hereda del componente gráfico *PictureBox* y se utilizará como lienzo donde se dibujarán los componentes gráficos.

Barrera: Clase que hereda de *SignalListenerGraphicComponent*permitirá controlar un componente gráfico.

PlantaSistema: Clase que hereda de *SignalListenerGraphicComponent*, es utilizada para la representación de una zona de carretera.

Semaforo: Clase que hereda de *SignalListenerGraphicComponent* y me permitirá controlar dos gráficos cuando el semáforo esté en verde y cuando se encuentre en ámbar.

Vehiculo: Clase que hereda de *SignalListenerGraphicComponent* y mostrará en todo momento donde se encuentra el vehículo hacia qué dirección va.

4.3Módulo Adquisición de Datos y Módulo Identificación RFID

La implementación software de los siguientes módulos se ha obtenido mediante la integración de la API proporcionada por los fabricantes, y gracias a ella se ha conseguido alcanzar los objetivos perseguidos en un menor tiempo que si lo hubiésemos desarrollado por nosotros desde cero. La API nos la han proporcionado bajo la plataforma C# .NET por lo que ha resultado muy fácil la adaptación con nuestra aplicación telemática.

La implementación que se ha realizado para el módulo de identificación RFID ha sido sencilla pues la API proporcionada por Motorola aportaba un ejemplo de su uso. A partir de dicho ejemplo hemos realizado nosotros la implementación en nuestra aplicación. En nuestro caso la tarea a desarrollar este módulo era sencilla pues sólo tenía que leer un único tag cuando la aplicación se lo indicase.

La implementación que se ha desarrollado para el control del módulo de adquisición de datos ha resultado costosa pues para su implementación primeramente se utilizó la API, pero tras darnos cuenta que el funcionamiento de la misma no era del todo el esperado, (no era posible hacer uso de la funcionalidad de eventos que traía la API) se implementó el controlador de la tarjeta adquisición de datos a partir de la comunicación básica que sí nos proporcionaba.

La forma de implementación ha sido incluir en nuestro proyecto las librerías proporcionadas por ambos dispositivos y a partir de ahí utilizar las clases que ésta nos proporciona. Para el uso de la tarjeta de adquisición se ha incluido: “AxInterop.ADVBUTTONLib.dll“,“AxInterop.AdvDIOLib.dll”y“Interop.ADVBUTT ONLib.dll” pero en la práctica sólo hemos llegado a utilizar “AxAdvDIOLib” pues las otras dos me proporcionan elementos gráficos que no he tenido la necesidad de emplear. Para el uso del lector RFID hemos agregado “Symbol.RFID2.Host” y utilizado las clases que ésta proporciona para obtener el control del Motorola XR480. Las librerías proporcionadas por Motorola no permiten controlar en la totalidad el dispositivo pero para este proyecto final de carrera solamente ha sido necesario implementar la parte de comunicación con los tags.

4.4 Bases de Datos

La base de datos utilizada en este proyecto ha sido MySQL. Es un sistema para la gestión de bases de datos. MySQL es un software de código abierto, licenciado bajo la GPL de GNU, es una alternativa a otros potentes sistemas gestores de bases de datos como son Oracle o Microsoft.

Las características más importantes de este sistema son la escalabilidad, estabilidad, seguridad de sus bases de datos y Soporta gran variedad de Sistemas Operativo. Permite trabajar en modo cliente servidor pero en nuestro caso trabajaremos en modo local, es decir, la base de datos se encontrará dentro del mismo PC donde se ejecute la aplicación.

Nuestro modelo sigue una estructura entidad-relación en la expresaremos sus entidades relevantes así como sus interrelaciones y propiedades.

Explicación de las tablas: En nuestro sistema podemos observar claramente dos objetos físicos: vehículos y tags. Para hacer el sistema redundante establecemos la relación de uno a muchos entre *vehículos* y *tags*. Otra tabla que observamos es la de *descargas* que será la acción que hagan un vehículo en nuestro sistema como también es probable que un vehículo realice muchas descargas hay una relación uno a muchos, como en el caso anterior, y como en nuestro sistema va a ver varios tipos de residuos que se tarifarán de manera diferente establecemos la misma relación uno a muchos con la tabla *descargas*.

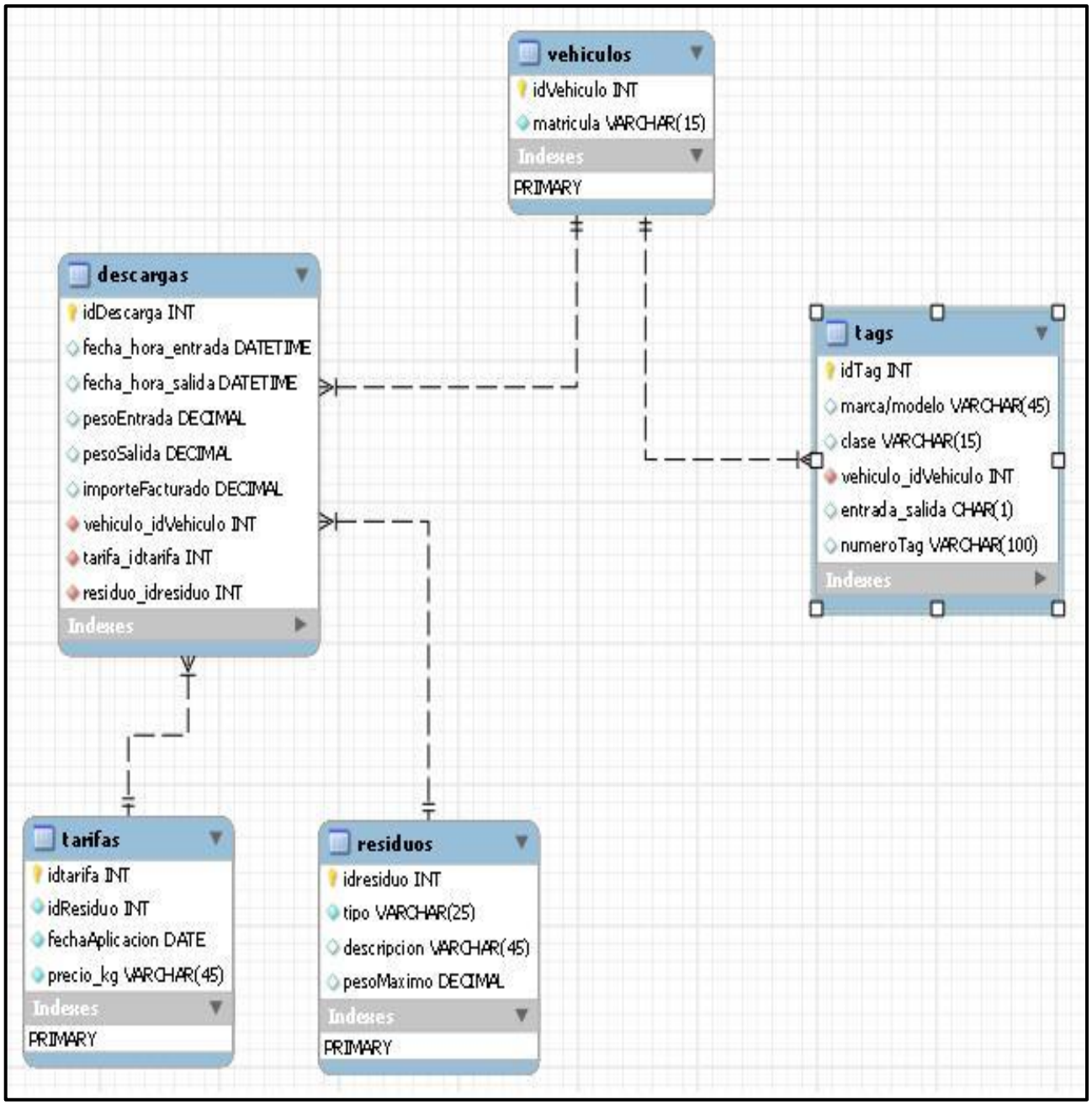

Figura 4.6: Modelo E-R y tablas

Campos de las tablas:

1. **Vehiculos:** contendrá todos los vehículos que pueden acceder al sistema.

    - **idVehiculo** : identificador interno del vehículo en la BBDD.
    - **matricula**: es la matrícula que ha de tener mi vehículo para entrar al sistema.

2. **Tags:** Encontramos parámetros que no hacen falta pero que han sido añadidos para una posible ampliación del sistema.

    - **idTag:** identificador interno del tag en la BBDD.
    - **marca/modelo:** indica la marca o modelo del tag.
    - **clase:** indica a qué clase de tag pertenece.
    - **entrada_salida:** indica si el tag está ubicado en la parte del camión que permita su identificación a la entrada o a la salida.
    - **numero_tag:** indica el tag que está registrado en el sistema.

3. **Descargas:** tabla que contempla todo el registro del sistema.

    - **idDescarga:** identificador de mi descarga.
    - **fecha_hora_entrada:** registro de entrada del vehículo al sistema.
    - **fecha_hora_salida:** registro de salida del vehículo al sistema.
    - **pesoEntrada:** peso del vehículo al entrar al sistema.
    - **pesoSalida:** peso del vehículo a la salida del sistema.
    - **importeFacturado:** importe facturado a la salida del sistema.

4. **Tarifas:**tarifas utilizadas en mi sistema.

    - **idTarifa:** identificador interno de la tarifa en la BBDD.
    - **idResiduo:** identificador del residuo aplicado a la tarifa.
    - **fechaAplicacion:** fecha en la que está operativa la tarifa.
    - **Precio_kg:** precio que vale el kilogramo del producto.

5. **Residuos:** los residuos de mi sistema.

    - **idResiduo:** identificador de la tabla residuo.
    - **tipo:** tipo del residuo.
    - **descripción:** descripción del residuo.
    - **pesoMaximo**: el máximo peso que se puede descargar del producto.

4.5 Lógica de control: Máquina de estados

Una vez que se tiene la visión general de la arquitectura se puede llegar a comprender en su totalidad la máquina de estados que se presenta a continuación, ya que esta máquina de estados relatará los estados en los que se puede encontrar el sistema durante el proceso, y si por algún casual ocurriese un fallo (error identificación, se dispara un sensor incorrecto, etc.) se tiene la posibilidad en la totalidad de estados de que la máquina informe al operario y por lo tanto no quede bloqueada. Se han de distinguir dos diagramas de flujo que representan dos comportamientos distintos de la máquina de estados; el primer diagrama mostrará el procedimiento que ha de hacer un camión a la entrada del sistema, mientras que el segundo mostrará el comportamiento del camión a la salida del sistema. Ambos diagramas convergerán a un mismo estado, el estado de *REPOSO*. Que ambos diagramas converjan a un mismo estado no es casualidad sino que se ha decidido hacerse así para otorgar al sistema estabilidad y escalabilidad, es decir, así permitimos que puedan entrar o salir de mi sistema dos vehículos consecutivos sin que sea necesario que el vehículo que ha entrado realice el proceso de salida.

A continuación vamos a explicar los estados del proceso de entrada de camiones.

Reposo: Estado inicial donde vamos a encontrar el sistema si no hay ningún vehículo en su interior. Este estado es el inicial tanto del procedimiento de entrada como el de salida. Para conseguir superar este estado hemos de fijarnos que el semáforo se encuentre en ámbar y además el sensor pulsado sea el FC1 como muestra el diagrama. Este sensor se puede apreciar en la Figura del esquema general eléctrico figura 3., en caso de que salte otro sensor pasamos al estado *Esp. OPERARIO*.

Identificación: Este estado posee redundancia, es decir, se va a comprobar por 2 vías. La primera identificación por RFID y la segunda comprobación será mediante una foto tomada a la matrícula. Si la identificación por ambas vías es correcta y el identificador del tag corresponde al de la matrícula detectada, entonces se pasará al estado "*Esperar camión* ". En el caso de que la identificación no sea válida porque falla uno de los dos sistemas de identificación o porque el id del tag detectado no se corresponda con la matrícula del vehículo en la *BBDD* pasaríamos al estado de *"ESP.OPERARIO"*.

Esperar Camión: En este estado se está esperando a que el camión se sitúe completamente sobre la báscula, esto es, quede posicionado entre las dos barreras. Mediante los sensores FC2 y FC3 detectaremos que el camión se encuentra en la posición correcta y se pasará el estado "*Tipo de Residuo"* en el caso de encontrarnos los dos estados .

Esperar Tipo de Residuo: Esta acción ha de realizarse en su totalidad por el usuario del vehículo, el cual ha de pulsar el pulsador para elegir el producto que transporta.

Esperar Pesada: Automáticamente en este estado se lanza la pesada del vehículo y se harán algunas comprobaciones para verificar que el producto elegido se encuentra entre los márgenes correctos de la pesada. Se dispone de un máximo de 3 intentos y en caso de excederlo se pasará al estado ESP.OPERARIO.

Esperamos Salida: En este estado la única información que aporta es que el vehículo se encuentra ya dentro de la planta de trabajo y no en la plataforma o sistema.

Esp. Operario: El estado operario es un estado inusual al que no se desea entrar. Este estado indica un posible fallo en la dinámica del transcurso del vehículo por el sistema. De este estado se saldrá mediante intervención del operario de cabina.

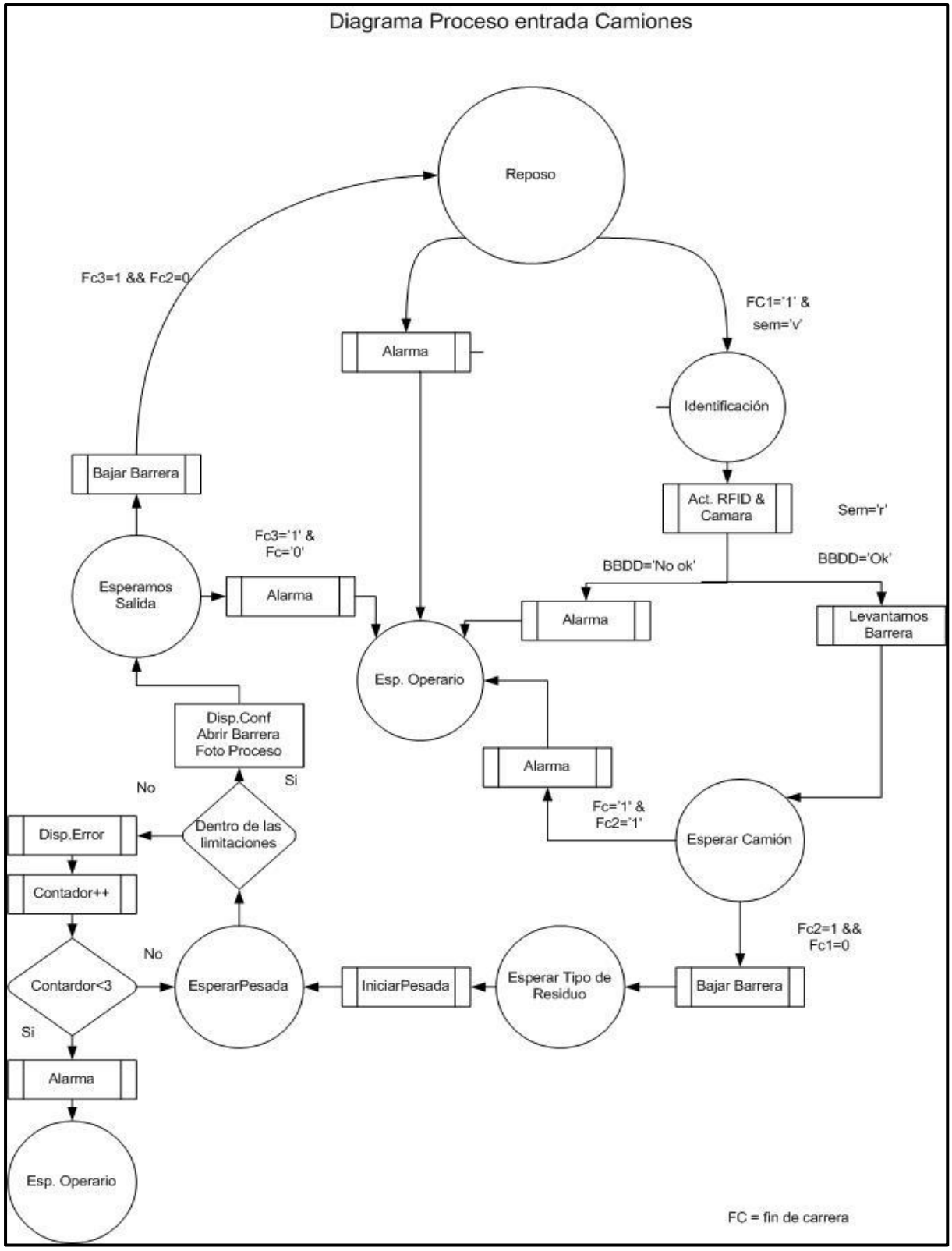

Figura 4.7: flujo grama a la entrada.

A continuación vamos a explicar los estados del proceso de salida de camiones cuyo diagrama se asemeja bastante al procedimiento de entrada.

Reposo: Estado inicial que comparten ambos procedimientos y que indica que el sistema se encuentra listo para ser usado, es decir, semáforos ambos en verde y barreras bajas y listo para iniciar el procedimiento de entrada o salida de un camión.

Detección: Estado en el que detectamos el vehículo que intentará salir de la planta. Automáticamente se activará la identificación y si el resultado de la misma es correcto si procederá a pasar al siguiente estado: "Iniciar Pesada". En este caso la identificación se hará solamente mediante RFID y no como en el procedimiento anterior que se hacía uso de la lectura de caracteres mediante una fotografía tomada si por casualidad el semáforo se encuentra en estado rojo saltaría una alarma avisando al operario de planta indicando que hay algo que obstaculiza el paso.

Iniciar Pesada: Para que el vehículo se encuentre en este estado ha de encontrarse correctamente situado tocando el sensor correspondiente. Si por casualidad no se encuentra correctamente situado se lanzará un timer y cada vez que pase la temporización se incrementara un contador cuando el contador alcance 3 intentos se procederá a llamar al operario para que solucione el conflicto en caso de que todo suceda convenientemente el conductor del vehículo recogerá el ticket y continuará.

Cierre Barrera de Salida: En este estado se procede a autentificar que no se encuentra ningún vehículo dentro de la plataforma observando los sensores en casa que los sensores estén en su estado de reposo pasaremos al estado de espera inicial para ello se bajaran las barreras y se colocaran ambos semáforos en verde entonces se habrá completado el circuito.

Esp. Operario: estado idéntico al del procedimiento de entrada. Estado inusual al que no se desea entrar este estado indica un posible fallo en la dinámica del transcurso del vehículo por el sistema de este estado se saldrá reinado la aplicación por parte del operario de cabina.

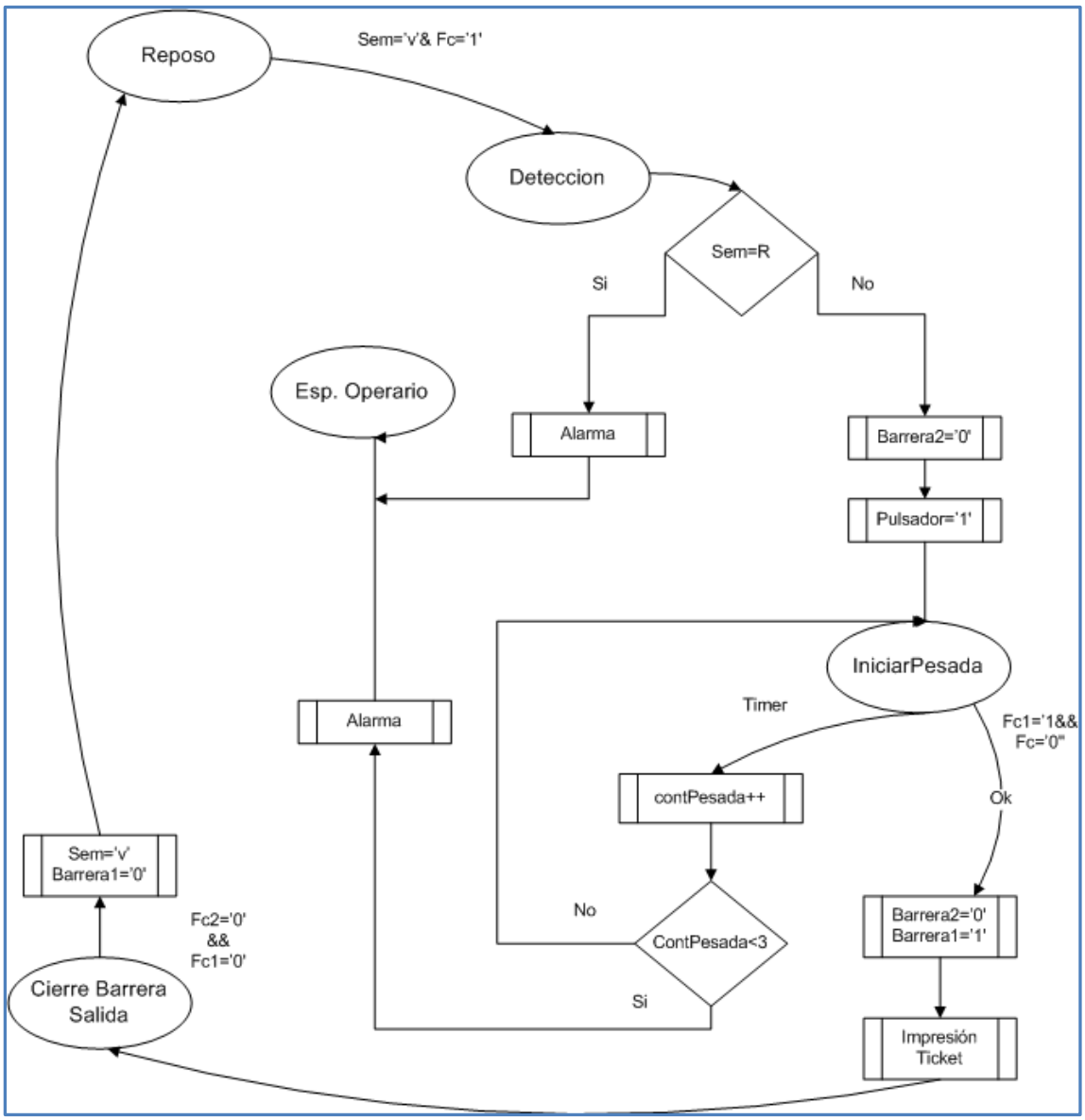

Figura 4.8: flujo grama a la salida.

A nivel software realizar la máquina de estados ha resultado una de las tareas más costosas del proyecto. Para su implementación se han utilizado 2 clases "*Estado.cs*" y "*Reglas.cs*" y un método "*ProcesarMaquinaEstados()*". Ahora se tratara de explicar el comportamiento de la parte lógica de la aplicación.

Se empezara por describir la clase Estado esta clase contendrá unas reglas asociadas al estado y un nombre que es utilizado como identificador, un estado puede contener una regla o más de una, pero en ningún momento puede haber un estado sin regla pues si se diese tal caso nunca se podría salir de dicho estado y esto conllevaría a que la aplicación quedaría bloqueada. Pues al no haber reglas que cumplir no habría cambio de estado.

Por otra parte nos encontramos con la clase Reglas que contiene los parámetros que se desean comprobar y los eventos que se han de lanzar una vez se ha cumplido la regla.

Parámetros:

- Comprobar mascara tarjeta adquisición.
- Comprobar pesada.
- Comprobar identificación.
- Comprobar display

En esta parte interviene el método *ProcesarMaquinaEstados().* Este método comprueba que la regla que contiene los parámetros que se deben de cumplir.En el momento que se comprueba que la regla es correcta se llama al método de la clase reglas que ejecutara las acciones que tenía marcadas lanzando los eventos.

Una vez que son lanzadas las acciones se pasará al siguiente estado que se pretende, es decir, si se ha comprobado que la regla del estado Reposo es correcta dependiendo la regla que haya resultado correcta se pasará al siguiente estado. En este caso si es la primera regla pasará al estado *Identificación* y si es la segunda regla pasará al estado *Deteccion*. En esta máquina de estados el único estado donde ha sido asignado mediante dos reglas ha sido el estado Reposo pues como se ha apreciado en los diagramas de flujos de este mismo capítulo es el único estado que comparten ambos procedimientos (procedimiento de entrada y procedimiento de salida).

4.6Administración y configuración del sistema

Debido a las características del sistema es necesario controlar diversos dispositivos de entre los cuales alguno precisa de un fichero de configuración. Que ha de cargar al inicio de la aplicación, para configurar sus parámetros internos. Este sistema de configuración se corresponde con el Lector RFID al que le proporcionamos un fichero de configuración estándar, es decir, bajo la extensión ".config", ya que este tipo de extensión era recomendado por la plataforma *.NET*, este archivo de configuración se encuentra alojado en un directorio en la raíz del disco duro y además para la estructura de este fichero se ha utilizado el lenguaje *XML* donde las etiquetas me indicaran los parámetros relevantes para la configuración. Éntrelos parámetros podremos encontrar puerto de comunicación, dirección ip, nombre del lector, etc.

Otra forma de administración en nuestro sistema es la referida al almacenamiento de los log. Los cuales son necesarios para recopilar información de nuestro sistema así como: funcionamiento, parámetros actualizados, estado del sistema, etc. Estos parámetros son guardados en un archivo llamado "log.txt" donde quedan registrados los estados y todas las entradas del sistema también se ha añadido en la interfaz gráfica un cuadro de texto a modo informativo y de sólo lectura donde en él irán apareciendo los eventos de la aplicación según vayan transcurriendo en tiempo real.

Para la administración de los datos que se han generado por la aplicación tal como imágenes de las matrículas y ticket. Se ha optado por crear un directorio para recoger dicha información, llamado imágenes, dicho directorio se encuentra en la carpeta del ejecutable y en él se distribuirán el contenido generado del siguiente modo. Para tener un mayor control de la gestión de los datos generados se ha optado por crear una carpeta con la fecha obtenida del sistema. Para que las imágenes de la matrícula que se vayan obteniendo queden guardadas en el sistema. Estas imágenes se irán guardando con el nombre de la hora que han sido tomadas. Así pues si en cualquier momento se desea recuperar las imágenes tomadas se dirigirán hacia la carpeta contenedora con la fecha del día en que el vehículo hizo esa tarea en la plataforma y la hora en la que ocurrió. Así se tendrá localizada la imagen. Para la administración de los tickets que se vayan generando, se optó por alojarlos conforme se fueran creando en el escritorio. Pues para su objetivo resultaba el sitio idóneo ya que se encontraban fácilmente al alcance.

4.7 Especificación de requisitos.

4.7.1 Interfaz Usuario.

Se ha de mencionar que no se ha prestado demasiada importancia a la creación de la interfaz gráfica que se ha desarrollado para la aplicación pues el presente proyecto final de carrera no tenía como objetivo desarrollar dicha parte. Si bien este proyecto final de carrera ha pretendido sacar una versión funcional de la aplicación a la que poder poner en marcha y detectar fallos que se hayan podido escapar en la fase de análisis.

La interfaz gráfica que se ha desarrollado cuenta con dos paneles. El panel principal o de configuración de parámetros en el que se presta atención a los parámetros referidos a cada uno de los dispositivos que se van a emplear en la aplicación. Dentro de este panel se puedes distinguir varias secciones, las cuales serán comentadas a continuación.

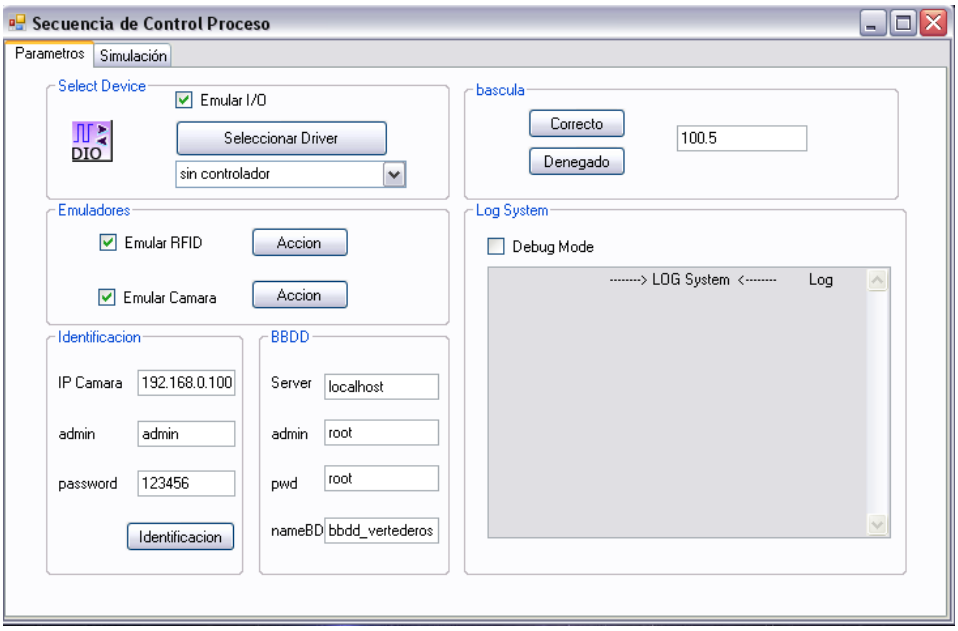

Figura 4.9: Panel principal

Sección *SelectDevice*, esta sección nos permite elegir entra un dispositivo la tarjeta adquisición de datos o el simulador de la misma. La sección Emuladores nos ofrece que la misma funcionalidad que la sección *SelectDevice*. En cuanto a la sección Identificación nos proporciona una serie de parámetros correspondiente al sistema de captación de imágenes que serán leídos de esta sección tal como usuario, password y dirección IP.

También tenemos una sección *Báscula* y cuya única función es la de recoger el dato que supuestamente nos proporcionará el dispositivo de pesada que como se ha comentado en capítulos anteriores todavía no se ha podido implementar. La última sección es *Log System* en ella irán apareciendo los eventos que transcurren en mi sistema presenta un *checkBox* para conseguir más información en los eventos.

En este panel vemos como se han realizados secciones de los contenidos dentro del mismo panel para lograr un mayor orden.

El panel secundario o de seguimiento de la simulaciones un panel cuya única finalidad es la de observar el estado del sistema en la aplicación mediante componentes gráficos que irán cambiando conforme transcurran las acciones dentro de la plataforma. Se dejará junto a este proyecto final de carrera un apéndice explicativo con el correcto uso de los simuladores en el que se podrán observar los cambios relativos realizados por la plataforma en este panel de seguimiento.

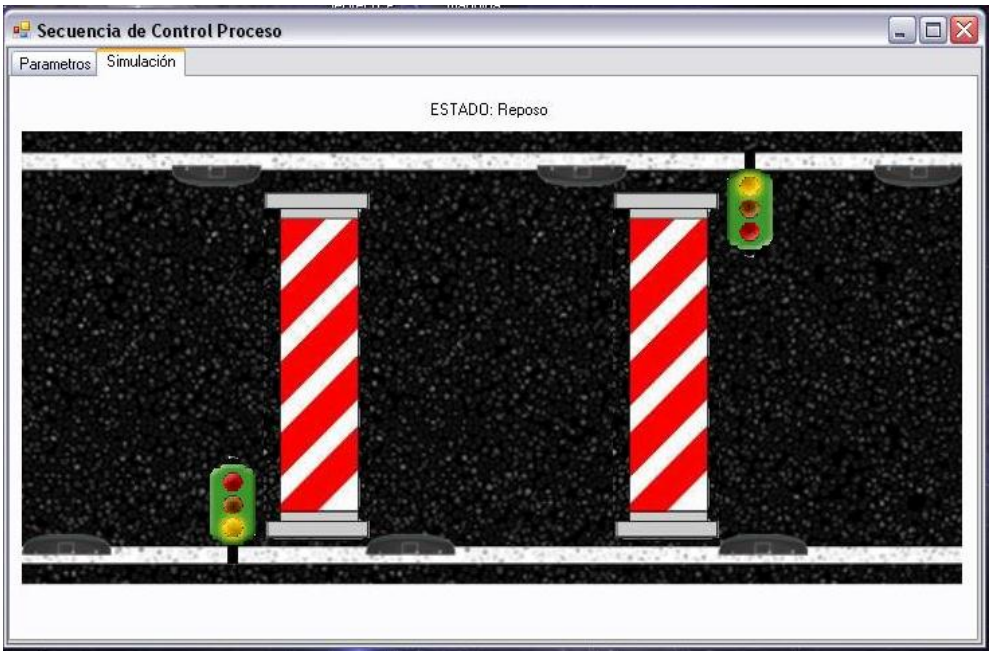

Figura 4.10: Panel secundario.

Este panel queda relegado a la observación y como se muestra en la figura 4.9 aparecen dibujados los componentes de la plataforma entre los elementos gráficos que encontraremos en el panel se encuentran , dos barreras, dos semáforos, un camión, planta del sistema, la cual simula una carretera con los sensores a las orillas.

4.7.2 Requisitos funcionales.

Son declaraciones de los servicios que proveerá el sistema, de la manera en que éste reaccionará a entradas particulares. En algunos casos, los requerimientos funcionales de los sistemas también declaran explícitamente lo que el sistema no debe hacer.

- El sistema debe registrar los vehículos.
- El sistema debe actuar frente a un correcto cambio en las entradas.
- El sistema debe guardar información de los vehículos (hora, peso, producto, etc.).
- El sistema ante cualquier error a de lanzar un aviso de fallo.
- El sistema ante un fallo en la identificación no debe permitir entrar nada.
- El sistema debe almacenar la información en un sistema aparte de la aplicación.
- El sistema ha de ser ampliable, escalable.
- El sistema ha de poder controlarse a distancia.
- El sistema debe de tardar el mínimo tiempo posible en hacer la identificación.
- El sistema ha de proporcionar un ticket al conductor del vehículo indicando algunos datos (fecha, hora, producto, cantidad, matrícula).

La mayoría de los requisitos funcionales han sido desvelados ya pero faltaría tratar un ínfimo detalle, la identificación. Resultaba primordial que el procedimiento mediante

el cual, el camión fuera identificado, denominado proceso de identificación, tardara el menor tiempo posible en realizar dicha tarea. Para ello se ha tenido que utilizar un recurso conocido en programación como *Thread* o en castellano hilos. Esta forma de programación nos permite desarrollar dos tareas casi simultáneamente, es decir, nos permite desarrollar una programación concurrente. La idea consiste en que cuando se ha de hacer la identificación redundante por medio de RFID y por la captura de la matrícula se lanzara ambos procesos en hilos diferentes para agilizar y reducir el tiempo en el que se lleva a cabo la identificación. De esta manera conseguimos dotar a la plataforma de una agiliza que nos permitirá tener el procedimiento de identificación como ventaja y no como un inconveniente como se ha visto en el capítulo 2.

Para poder atender al requisito funcional de la aplicación para entregar ticket al conductor. Ha de obtener los datos de su paso por la plataforma para ello se ha utilizado una librería de código abierto (Open Sourcelibrary). Llamada PDFsharp, con esta librería se han conseguido desarrollar fácilmente un formato básico de impresión de ticket para algunos datos de la base de datos. Una vez el vehículo ha llevado a cabo los dos procedimientos el de entrada y de salida.

4.7.3 Requisitos no funcionales

No se refieren directamente a las funciones específicas que entrega el sistema, sino a las propiedades emergentes de éste como la fiabilidad, la respuesta en el tiempo y la capacidad de almacenamiento. A continuación se presentaran algunos de los requisitos no funcionales.

FUNCIONALIDAD

Aunque todavía no se ha entregado una versión final da la aplicación. Si se puede hablar en cierta medida da la funcionalidad de la aplicación. En esta medida se puede apreciar que la aplicación presenta una división en el panel de configuración para facilitar la experiencia con el usuario. Donde cada dispositivo se configurará con un bloque de parámetros, es decir, se aprecia claramente la distinción entre los bloques de parámetros de cada dispositivo y esto permite que le usuario de la aplicación con un simple golpe de vista e intuitivamente deduzca el uso de los parámetros. Debido a las características de la aplicación. Puesto que no es una versión final del desarrollo se ha optado por no incluir una contraseña al inicio de la aplicación por lo que la seguridad en ese aspecto que presenta la aplicación es inexistente.

CONFIABILIDAD.

Entendiendo confiabilidad como la madurez de la aplicación en la que debe presentarse información al usuario. Sobre los errores que comete al utilizar la aplicación. Estos errores deben estar bien identificados y en el idioma Español. Los mensajes de error deben contar como una ayuda para orientar al usuario en su trabajo y así no cometer reiteradamente el mismo error, además debe existir una explicación del porqué del error esta sección se tiene bien cubierta mediante los log a los que el usuario puede hacer una consulta siempre que lo necesite. Además cada vez que ocurre un error se le notifica al usuario. Pues como se puede apreciar en ambos diagramas de flujo mediante el estado Esp. Operario este estado está encargado de la comunicación de errores.

FACILIDAD DE USO.

Esta quizás sea el requisito al que más atención se le ha prestado. Ya que resulta vital que para hacer uso de la aplicación no se necesite realizar ningún curso de formación en la aplicación, es decir, que cualquier persona con unos mínimos conocimientos de informática sea capaz de darle un buen uso a la aplicación. Dentro de esta sección se pueden encontrar 3 puntos.

- La Facilidad Cognoscitiva.

Se considerará la facilidad de comprensión. El panel de simulación con imágenes sirve para una mejor comprensión y aprendizaje de la aplicación, tomando en cuenta el desarrollo de la aplicación en tiempo de ejecución.

- La Facilidad de Comprensión.

La aplicación ayuda al usuario en la tarea de supervisión que desarrolla pues mediante el panel de simulación le muestra el estado de la plataforma mediante gráficos. Se explicar de una manera correcta las distintas opciones que le permiten la aplicación o una determinada interfaz.

- La Atracción.

No se define un estándar de interfaz tomando en cuenta los colores de la institución y los colores de la empresa que desarrolla la aplicación todavía. Ya que no nos encontramos aún en una fase final del desarrollo de la aplicación. Pero teniendo en cuenta esta consideración se tiene la probabilidad de que el usurario este cómodamente trabajando con la aplicación.

EFICIENCIA.

Atendiendo a la utilización de los recursos. Se utilizan procedimientos básicos, que no tengan más de una sentencia, esto es beneficioso para el acceso a la base de datos. Los filtros de búsquedas o lógica del negocio para los datos tratando de evitar que el acceso a la base de datos sea el cuello de botella.

PORTABILIDAD.

- La facilidad de adaptación.

Se ha utilizado un marco de trabajo que garantiza un mantenimiento respecto a la tecnología que pueda aparecer y a los paradigmas de desarrollo de interfaz, es decir, no hemos apoyado en la versatilidad de C# para el desarrollo Además, la interfaz a desarrollar debe ser probada en distintos escenarios posibles de computadoras, para de esta forma garantizar la adaptación de la aplicación.

- La facilidad de Instalación.

Se ha de crear para la aplicación un paquete de instalación con su respectivo manual para que los encargados de la aplicación puedan restablecer con bastante rapidez la aplicación, debe tomarse en cuenta que pude ser más de un instalador ya que la aplicación está orientada a los servicios. Solo se tomará en cuenta la tecnología Windows para desarrollar el paquete de instalación.

- La coexistencia.

Se podrá instalar en una sola computadora toda la aplicación, ya sea los sub-servicios que presenta como la propia aplicación. Pero, deben tener características independientes.

4.8 Implementación y uso de la aplicación.

4.8.1 Implementación.

En la fase de implementación de un proyecto de un sistema informático se construye una solución software utilizando las tecnologías y herramientas seleccionadas en la fase de diseño. A continuación se va a hablar sobre la metodología que se ha seguido para dar forma a todo esto que se está comentando aquí.

Para el desarrollo de cada una de las implementaciones se optado por atacar los problemas de lo más sencillo a lo más complejo, es decir, para integrar la tarjeta adquisición de datos en la aplicación. Lo primero fue desarrollar pequeñas aplicaciones para poco a poco ir subiendo en complejidad y terminar llegando a la aplicación funcional. Que finalmente estará integrada en la aplicación telemática. Este mismo procedimiento ha sido empleado para incluir el resto de dispositivos en la aplicación. Por ejemplo para el control del RFID primeramente se desarrolló una aplicación que tuviera comunicación con el *Reader*. Apto seguido se desarrolló una aplicación para la lectura de tag.Y se implementó en la aplicación telemática.

Mediante esta técnica hemos conseguido ahorrarnos tiempo en el desarrollo de la solución final de la aplicación. Pues una vez que se tenía dominado un dispositivo y se sabía perfectamente el rendimiento del mismo se podía encajar la funcionalidad en el marco de trabajo que se había establecido.

La metodología de trabajo seguida para el desarrollo del software de este proyecto, que consistió fundamentalmente y debido a lo reducido del grupo de trabajo, (El responsable de la sección de programación de Digital-Home S.Coop, D. Antonio Martínez y el alumno en prácticas Miguel Alarcón) en una serie de reuniones para desarrollar la aplicación telemática, de manera que se recogieran todas las características y funcionalidades deseadas para la creación de la aplicación de forma satisfactoria. Y en una revisión de los incrementos conseguidos, como soluciones parciales de la funcionalidad de la aplicación.

En esta parte de la memoria no se va a realizar, debido a la cantidad de espacio que ocuparía, y por motivos de privacidad, una descripción minuciosa del código utilizado para configurar la funcionalidad de la aplicación. Lo que sí que se va a comentar a continuación es la funcionalidad de la aplicación desarrollada en el apartado 4.8.2 Uso.

4.8.2 Uso.

En la aplicación se pueden hacer dos grandes distinciones referentes al uso de la aplicación telemática:

La primera resultaría de ejecutar la aplicación con todos los equipos conectados, es decir, sin la necesidad de tener los simuladores de los equipos ejecutándose. En este caso la aplicación quedaría relegada meramente a informar por el cuadro de log o mediante de incidencias de la plataforma de cómo se encuentra el sistema. Y por medio de la pestaña de simulación con la parte gráfica. Esta sección no nos aporta más información de la ya obtenida con el log. En esta parte se vería el estado del sistema gráficamente y también nos saltarían aquellas alertas que tenemos contempladas actualmente con las que informar al operario de Planta.

La segunda gran distinción que se puede hacer acerca del uso de la aplicación. Es la de hacer uso en su totalidad o en parte de los simuladores desarrollados para el presente proyecto final de carrera. Donde en el panel de configuración se encontraría los campos para configurar el acceso a la bbdd, parámetros como servidor, admin, contraseña, nombre de la Base de datos.

Otro de los campos es el de identificación mediante el recocimiento de caracteres cuyos parámetros son ip, admin y contraseña del dispositivo. Podemos observar otro campo que es el que me muestra por pantalla la escritura de los log en tiempo real y también un campo que me permite emular la báscula indicando como parámetro si el peso es correcto o no y la cantidad pesada.

El otro panel es el de simulación donde lo que podemos observar es el estado de nuestro sistema gráficamente incluyendo donde se encuentra el camión el color de los semáforos y el estado de las barreras que cambiará todos los elementos gráficos dependiendo del estado en que se encuentre mi sistema Esta ventana no aporta ninguna información adicional.

Capítulo 5: Análisis Financiero.

En este capítulo se procederá a realizar un breve análisis acerca de la inversión llevada a cabo para poner en marcha el presente proyecto. En este capítulo se verá el coste aproximado de los material también se verán los tiempos de instalación y puesta en marcha de la plataforma además se realizarán unos cálculos para poder observar los beneficios económicos que representa ejecutar el proyecto.

5.1 Adquisición e instalación de los equipos.

En este apartado se detallará el coste económico en la adquisición de los equipos, material y la instalación de los mismos.

PROYECTO

Descripción	Ctd.	Precio un.	Total	PRECIO DHBT
Sistema RFID 2 antenas 2 lectores 2 cables 1 Fuente de alimentación 1 Soporte	4	1885,6	7542,4	10559,36
Tags para camiones	150	5,5	825	1155
Tags invitados	20	5,5	110	154
Sistema de detección (4 sensores + peanas)	16	160	2560	3584
Tarjeta de adquisición de datos	4	160	640	896
Barreras	4	1400	5600	7840
BOTONERA SELECCIÓN DE RESIDUO Botoneras Display Luces Portero	4	1600	6400	8960
BOTONERA TICKET	4		0	0
Semáforo	8	200	1600	2240

Grupo electrógeno	4	940	3760	5264
Sistema de alimentación ininterrumpida	4		0	0
SERVIDOR Servidor Monitor Teclado Ratón Impresora USB	4	2560	10240	14336
MODEM GPRS	4	240	960	1344
OBRA CIVIL	4	2400	9600	13440
DESARROLLO DE LA APLICACIÓN SOFTWARE			14500	20300

-: en el listado se inlcuyen los precios del equpiamiento en su forma "neta".
No se han cosiderado para ninguno de ellos, ningun tipo de incremento ni margen comercial.

TOTAL:	64337,4	90072,36
BENEFICIO:		25734,96

Asumiendo los costes de adquisición e instalación nos representa un total de:

Adquisición	Instalación	Total
108987,55	8222,5066	117210,06

5.2 Diagrama de tiempos de la instalación.

En este apartado se expondrá el diagrama de gannt del montaje de la plataforma previamente cumplidas las etapas anteriores de ingeniería así como los tiempos necesarios para llevarse a cabo. Se ofrece una explicación del desarrollo del montaje de la plataforma.

RED DE COMUNICACIONES	Duración	Mes 1	Mes 2	Mes 3	Mes 4
1. REPLANTEOS Y TRABAJOS PREVIOS	**2 semanas**				
1.1 Revisión de acometida eléctrica.	2 semanas				
1.2. Revisión y confirmación obra civil	2 semanas				
2. DESARROLLO DE APLICACIÓN DE CONTROL	**3 meses**				
2. OBRA CIVIL	**1 mes**				
2.1 Realización de la obra civil	3 semanas				
2.2. Acometida eléctrica	2 semanas				
3. ACOPIO DE MATERIALES Y EQUIPAMIENTO.	**4 semanas**				
3.2. Sistema de identificación por RFId	2 semanas				
3.3. Barreras, cuadros de control,etc.	2 semanas				
3.4 Equipamiento informático	1 semana				
4. INSTALACIÓN Y CONFIGURACIÓN.	**1 mes**				
5. AJUSTE Y PUESTA EN MARCHA.	**1 mes**				
6. FASE DE PRUEBAS	**1 mes**				

Atendiendo a la necesidad impuesta por el cliente de que la planta en ningún momento cesara su actividad. Se ha decidido hacer la implantación de la arquitectura por módulos, es decir, si se procede a instalar el equipamiento informático punto 3.3 del diagrama de tiempos en la plataforma. Ha de hacerse evitando en la medida de lo posible invadir o molestar el trabajo realizado por los operarios de planta. En cuanto al desarrollo de la aplicación software se han contemplado 3 meses de trabajo, para su desarrollo. Además se ha contemplado un mes de ajuste y puesta en marcha en el que se encajara la aplicación y la plataforma para que efectúen un correcto funcionamiento.

5.3 Beneficios económicos tras la implantación del proyecto.

En este apartado se justificara mediante el beneficio económico el porqué de la ejecución del proyecto.

Una de las principales razones por las que cualquier cliente se ha de decidir a actualizar su sistema de trabajo es probablemente la pérdida de ingresos en su facturación. Debido a fallos mencionados en el capítulo 2 del presente proyecto final de carrera. En capítulo 2 se documentaban algunos de los problemas que existen actualmente en el transcurso de la actividad de una planta genérica. Uno de los problemas era la perdida de información en la facturación, lo cual conllevaba a la pérdida de capital por parte del cliente. Puesto que no se podrían emitir todas las facturas.

Los ingresos de los que puede subsistir una planta de reciclaje de cualquier producto, es decir, una planta genérica son pesaje más el uso de la planta. Con la inversión en el presento proyecto final de carrera lo que se pretende es mediante la automatización de una planta de reciclaje genérica obtener un mayor control de los pesajes llevados a cabo por la planta. Y así minimizar los errores llevados a cabo en la facturación.

Ahora se pretenderá explicar mediante una hipótesis el margen de beneficio que se conseguirá implantando el sistema en una planta genérica. Si actualmente un camión hace una media de 25 pesajes/camión en un mes y tenemos registrados en el sistema alrededor de 30 camiones. Con el actual sistema de trabajo es habitual que se produzcan alrededor de 3 errores en los pasajes por camión en un mes. Implantando el presente proyecto final de carrera se conseguiría bajar dicha tasa de error alrededor de 1 error ó 0.
Resumiendo:

- Hay un total de 30 camiones registrados.
- Con unos pesajes en media de 25 pesajes/camión en un mes.

  Sin plataforma ---------- 3 errores
  Con plataforma ---------- 1 error

Con el sistema actual de trabajo se producen en las instalaciones 30 camiones x 25 $\frac{pesajes}{camion}$ = 750 pesajes en un mes. De los cuales se producen 3 $\frac{errores}{camion}$ x 30 camiones = 90 errores de los pesajes en un mes y por tanto de facturas que no se van a poder emitir. Esto es un 12% de la facturación total en un mes que no se puede emitir y por tanto cobrar. Con el sistema con plataforma se consigue bajar en torno a 1 error o menos en el peor escenario posible siendo 0 el mejor escenario posible. Es decir, solamente se cometerían 30 errores en el total de los pesajes realizados en un mes con lo que tendríamos unas pérdidas en la facturación del 4%. Es fácil comprobar que nuestra facturación implantando el proyecto aumentaría entorno al 8 % o incrementándose al 12% de no existir ningún error, el caso más favorable.

A continuación su planteará la hipótesis del retorno de la inversión del proyecto por parte del cliente. Los beneficios obtenidos por la plataforma al año y asumiendo la reubicación de recursos humanos, es decir operarios de planta, impulsados por la implantación del proyecto. Se puede catalogar como X, siendo X unidad monetaria y situándonos en el escenario más favorable. Se nombrará al valor monetario del

desarrollo del proyecto Y. Suponiendo que solo se destinara el beneficio generado por el proyecto, es decir, 12%X al pago del desarrollo del proyecto o lo que es lo mismo Y. Tomando números hipotéticos quedaría X = 180.000 € e Y = 117.210,06 €. Destinando el 12 % de los 180.000 € de facturación serían 21.600 € anuales. El proyecto quedaría amortizado por el cliente en apenas 5,5 años sólo invirtiendo el dinero recaudado de la mejora llevada a cabo por el proyecto.

Capítulo 6: Conclusiones y líneas futuras.

6.1 Conclusiones.

Durante la realización del presente proyecto fin de carrera se han cumplido los objetivos principales y se han establecido los cimientos y las bases para continuar con el desarrollo de un proyecto de tal característica para implantar el proyecto en un escenario real. Los principales hitos conseguidos son:

- **Estudio y redefinición de todos los procesos operativos de las plantas de reciclaje.**

  Identificación de los procesos actuales.
  Redefinición de los procesos.
  Comparación con las mejoras propuestas en la redefinición.

El primer objetivo marcado fue la realización de un completo estudio del anterior modelo que seguían los vehículos cuando realizaban sus tareas en una plataforma de estas características, y este se ha concluido con éxito. El estudio ha sido clave para obtener una visión clara sobre los procedimientos actuales de trabajo, localizar los puntos débiles y poder empezar a visualizar una solución completa de las necesidades así también ha servido para redefinir los flujos de trabajo y actualizarlos a los tiempos que corren.

- **Desarrollo de una maqueta para la simulación de la planta de reciclaje.**

  Conexionados de los componentes electrónicos de la maqueta.
  Conexión de los equipos a utilizar RFID, DAQ.

Lo más significativo, en esta parte del proyecto ha sido la toma de contacto con los materiales utilizados que personalmente han servido para afianzar varios conceptos relacionados con la electricidad y electrónica que se han adquirido en el transcurso de los años de carrera. Además también ha servido para realizar la puesta en marcha de la aplicación para su depurado.

• Diseño, desarrollo y puesta en marcha de la aplicación.

Se ha diseñado una arquitectura que es modular y escalable. Se han utilizado las herramientas ofimáticas que ya se disponían en Digital-Home S.Coop permitiendo la transparencia y facilidad del sistema. En cuanto a la puesta en marcha decir que el proyecto se encuentra aún en fase de desarrollo, para poder ser implantado el sistema en un campo real primeramente han de realizarse más pruebas en la maqueta desarrollado y que todas concluyan con éxito para poder adoptar la solución.

En resumen, se concluye que la nueva aplicación desarrollada con motivo de este proyecto final de carrera, cumple con la mayoría de los objetivos planteados al inicio del mismo, y que se ha creado un entorno colaborativo basándose en la reingeniería de los procesos, que hará mucho más fáciles, a la vez que potentes, todos los procesos operativos que tienen lugar en una plataforma para el control de entradas y salidas de camiones

Además se ha cumplido con el objetivo de añadir una serie de servicios de valor añadido, que hacen que esta aplicación cuente con una serie de funciones extra, que mejoren el trabajo y se lo faciliten al utilitario de la aplicación.

6.2 Líneas Futuras.

La actual versión de desarrollo que se alcanzado en el presente proyecto final de carrera queda lejos de la solución que se adoptara al final de proceso productivo pero debido a la metodología seguida por la empresa lo primordial era una primera versión a la que hacerle pruebas en la maqueta que se desarrolló, por ello es fácil de observar que hay distintos módulos que se han tenido en cuenta pero que aún quedan por implantar.

Uno de estos módulos puede ser el de pesada. La cual se ha tenido en cuenta en el desarrollo, es decir, se ha incluido en el desarrollo software las partes relevantes a este módulo simulando el dispositivo en el panel de configuración pero dado que no se tenía conocimiento del tipo de dispositivo que se iba a utilizar para esta tarea se opto por no avanzar más con el desarrollo.

Otra funcionalidad que no se consideró prioritaria era la referida al display. Esta funcionalidad ha corrido la misma suerte que el módulo de pesada. Ambas funcionalidades sería interesante implementarlas y observar el comportamiento de la aplicación.

Otra posible mejora sería la de controlar la aplicación no desde el propio PC que la ejecuta si no que a distancia, es decir, que se pudiera tener cierta información de cómo se encuentra el estado de la plataforma sin tener que estar delante de la aplicación.

Todos estos posibles cambios son el resultado del gran auge de las tecnologías de la información y uso que se están dando por parte de los usuarios. Por todo esto y porque cada vez más la gente utiliza este tipo de tecnologías al alcance de muchos, este proyecto podría estar en continua evolución.

Capítulo 7: Bibliografía.

7.1 Referencias.

- **http://www.motorola.com/Business/US-EN/Product+Lines/Symbol**
  Información acerca de la tecnología RFID implementada en el presente proyecto.

- **http://www.advantech.com/products/USB-4761/mod_C1E301AB-CDC8-45C0-B610-6AEA44B544AE.aspx**
  Información acerca de la tarjeta DAQ.

- **http://www.monografias.com/trabajos34/base-de-datos/base-de-datos.shtml**
  Diseño de base de datos.

- **http://msdn.microsoft.com/en-us/library/aa288436(v=vs.71).aspx**
  Liberáis de C Sharp y ejemplos.

- **http://www.devjoker.com/contenidos/C/158/Llamadas-asincronas.aspx**
  Uso de hilos en C Sharp

- **http://support.microsoft.com/**
  Ayuda y Soporte Técnico de Microsoft España

- **http://www.degerencia.com/tema/reingenieria_de_procesos**
  Definición e información complementaria de la reingeniería de

A.Anexo – Manual de Usuario de los simuladores.

El presente documento tiene como objetivo servir de apoyo y proporcionar a los usuarios los conocimientos necesarios para el uso efectivo de los simuladores en la aplicación telemática.

En este documento se hará una presentación de los simuladores desarrollados y se procederá a hacer una demostración dándole un buen uso a los mismos. En los siguientes párrafos se expondrán los simuladores y las características que poseen.

Emulador tarjeta DAQ

Ilustración 1: Simulador DAQ

Este simulador es el encargado de sustituir a la tarjeta adquisición de datos en su labor. Mediante la cual se simularan todos los elementos activos de la maqueta o del escenario real. Como se puede apreciar fácilmente el simulador estará compuesto por 8 entradas y 8 salidas digitales. A la izquierda del simulador se encuentran las salidas con las que se manejaran los siguientes elementos: semáforo, barrera y la confirmación de elección de producto. A la derecha se pueden encontrar las entradas, mediante la cuales controlaremos los finales de carrera que me posicionaran el vehículo y los pulsadores.

Como se puede apreciar, es fácil distinguir el nivel lógico que posee cada entrada pues para hacerlo más intuitivo se configuraron las entradas con color verde a nivel alto y color rojo nivel bajo.

Para cambiar de estado una entrada basta con elegir la entrada desea y hacer clic encima de la misma con ello conseguiremos que cambie a su estado opuesto, es decir, si se encuentra en un nivel alto se pasará a un nivel bajo y si se encuentra en un nivel bajo se

pasará a un nivel alto. En las salidas se ha utilizado un recuadro indicando el nivel lógico, 0 cuando se encuentre a nivel bajo y 1 cuando se encuentra a nivel alto.

NOTA: Como se puede apreciar las siete primeras entradas se encuentran de color verde o lo que es lo mismo a nivel alto. Esto sucede porque en el desarrollo de la maqueta los finales de carrera se comportan como NC (normalmente conectados). Se pensó así porque resultaría más fácil detectar si un final de carrera no funcionase correctamente.

Emulador sistema RFID

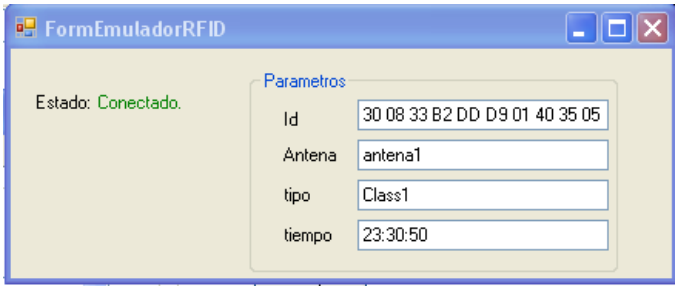

Ilustración 2: Simulador RFID

Este simulador es el encargado de sustituir los componentes hardware RFID de la aplicación y así permitir emular un sistema de RFID completo (reader, antena, tag). Nada más se arranca el simulador las primeras acciones que se llevan a cabo es la de establecer su estado como conectado y recoger la hora del sistema.

En la parte superior izquierda se puede observar el estado en el que se encuentra el reader. Se observará con el texto de conectado y en color verde cuando el simulador realice la tarea y se observará con el texto de desconectado y de color rojo cuando este no tenga conexión. También se puede encontrar a su derecha una serie de parámetros emulando un tag, es decir, cuando se lance el proceso de leer un tag los valores de los campos: id, antena, tipo y tiempo será de donde se recoja la información como cuando el lector recoge la información de un tag en un caso real.

Para interactuar correctamente con este simulador basta con rellenar con los valores correcto los parámetros. Introduciendo un tag que este dado de alta en la base de datos, indicando la antena que lo ha leído mediante el parámetro antena y el tipo de tag mediante el parámetro tipo. El parámetro de tiempo no hace falta especificarlo pues cuando arranca el simulador automáticamente recogerá la hora del sistema. Una vez introducidos los valores correctos en los parámetros y cuando se inicie el proceso de identificación estarán listos. Y serán tomados del simulador.

Emulador de la del dispositivo de captación de imágenes

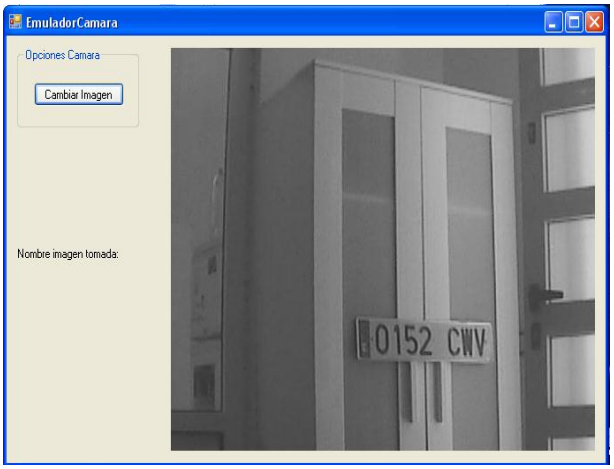

Ilustración 2: Simulador dispositivo de captación de imágenes

Este simulador es el encargado de sustituir la funcionalidad de un dispositivo encargado de hacer capturas de imágenes simuladas. Con la diferencia de que el dispositivo real ha de tomar una imagen en tiempo real. En este simulador se cargará una imagen que será mostrada en todo momento. Cuando sea necesario hacer uso del dispositivo. Al programa le será pasada la imagen mostrada en el simulador. La imagen mostrada en el simulador puede ser cambiada mediante el botón llamado "*Cambiar Imagen*". Este botón abrirá una ventana *OpenFileDialog* en la que sólo será capaz de ser cargados algunos tipos de imágenes.

Simulación

Ahora se procederá a explicar el uso de los simuladores y como llevar a cabo una simulación satisfactoria.

Se arrancaran los simuladores arriba mencionados indicando con el *checkBox* que se desean simular y se hará clic en sus botones correspondientes.

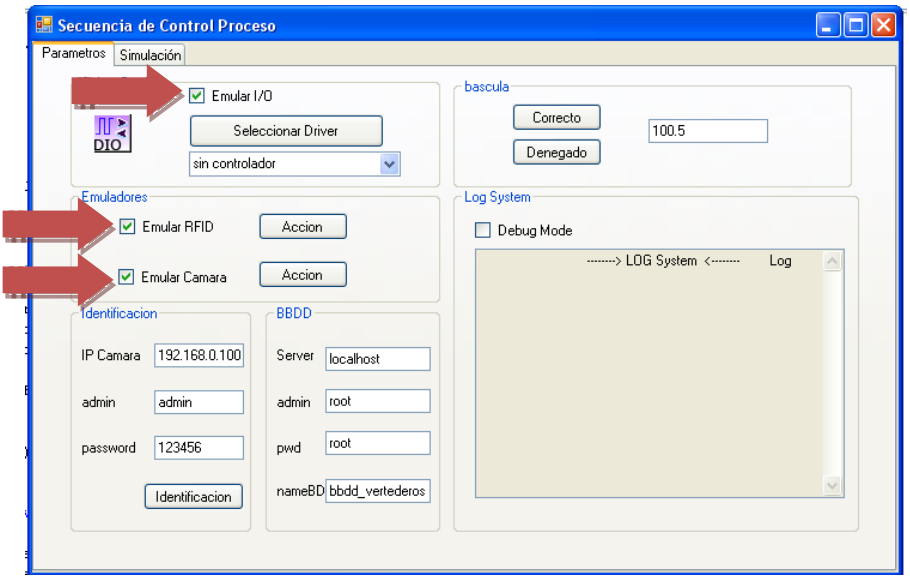

Ilustración 3: Elección emuladores

Una vez que han sido arrancados todos los simuladores se estará preparado para realizar la simulación. Debido a los simuladores desarrollados y a la usabilidad de los mismos solamente se interactuará con la tarjeta DAQ en concreto actuando sobre sus entradas. Los otros simuladores (RFID, dispositivo de captación de imágenes) serán utilizados con los valores que traen por defecto.

Para comenzar la simulación se excitara la primera entrada.

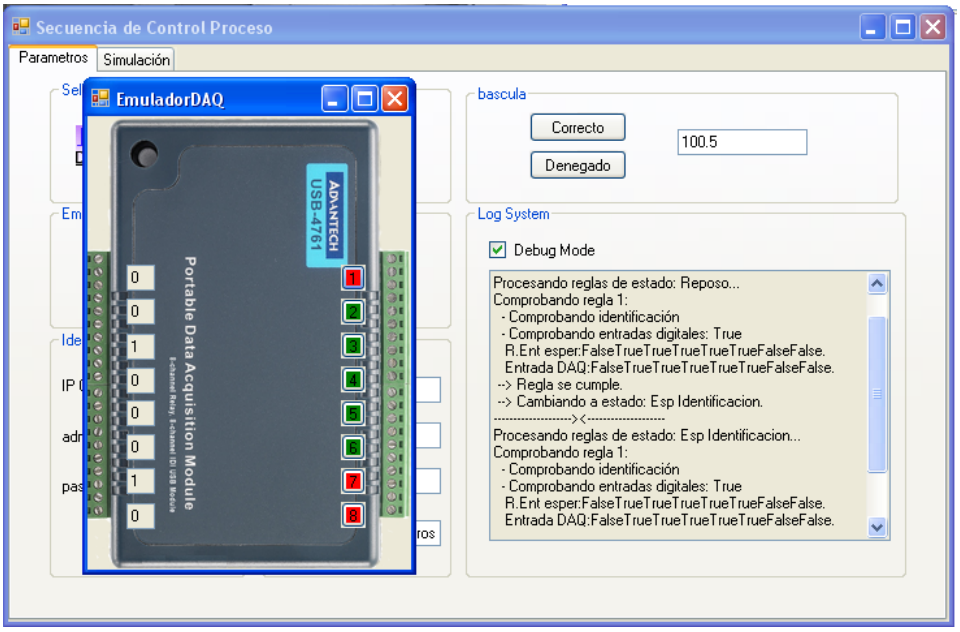

Ilustración 4: Simulación paso 1

Se puede observar que se ha producido la detección del vehículo y automáticamente se ha lanzado la identificación con el estado *Esp. Identificación*. En este estado han intervenido los dos simuladores restantes que han sido desarrollados (RFID, dispositivo de captación de imágenes).

La siguiente acción lógica que ha de suceder es que el camión deje de excitar el final de carrera 1 y pase a excitar el final de carrera siguiente entonces el camión se encontrará situado entre ambas barreras, se encontrará en la zona de la bascula.

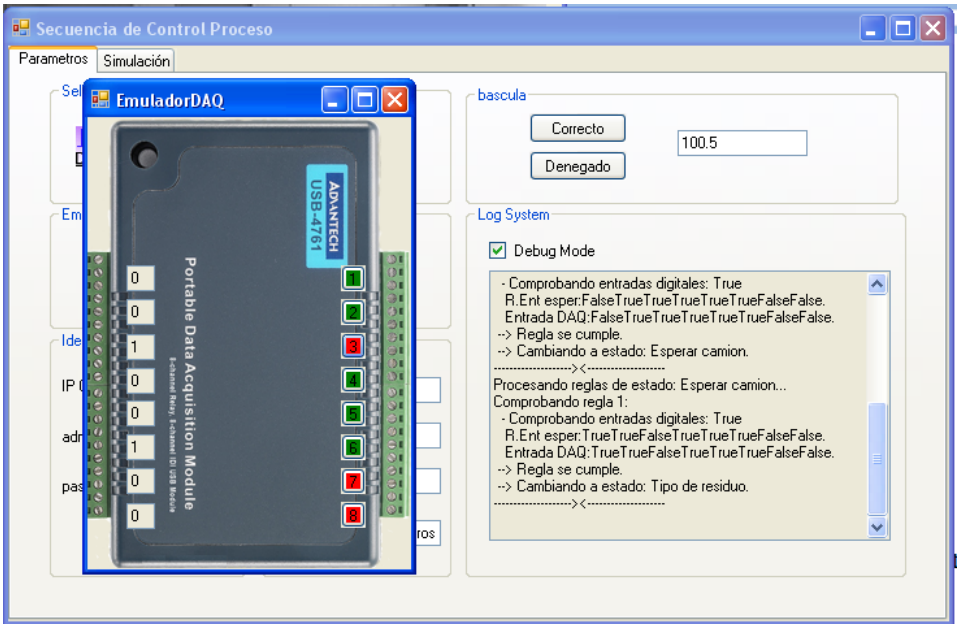

Ilustración 5: Simulación paso 2

Nota: debido a un error en la conexión de los cables en la maqueta el simulador se desarrollo cambiando los finales de carrera 2 por 3 y viceversa.

Actualmente el sistema se encuentra esperando que el conductor elija el producto. Para ello el conductor ha de presionar la botonera. El botón de la botonera que se corresponde con la entrada en la tarjeta adquisición de datos es el número 8.

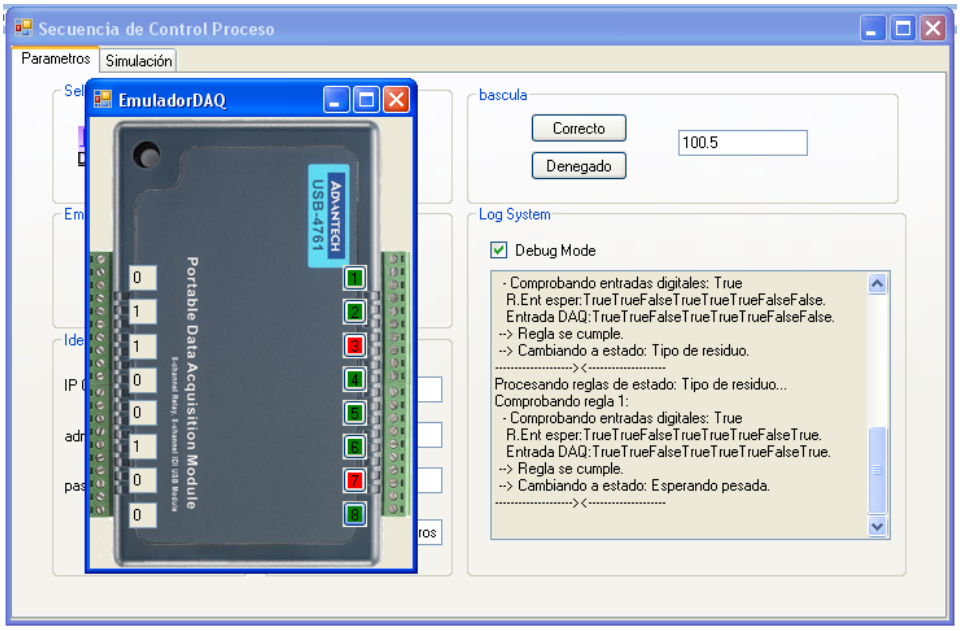

Ilustración 6: Simulación paso 3

Actualmente el sistema se encuentra esperando que el camión actué sobre el ultimo final de carrera y así el sistema volverá a al estado reposo.

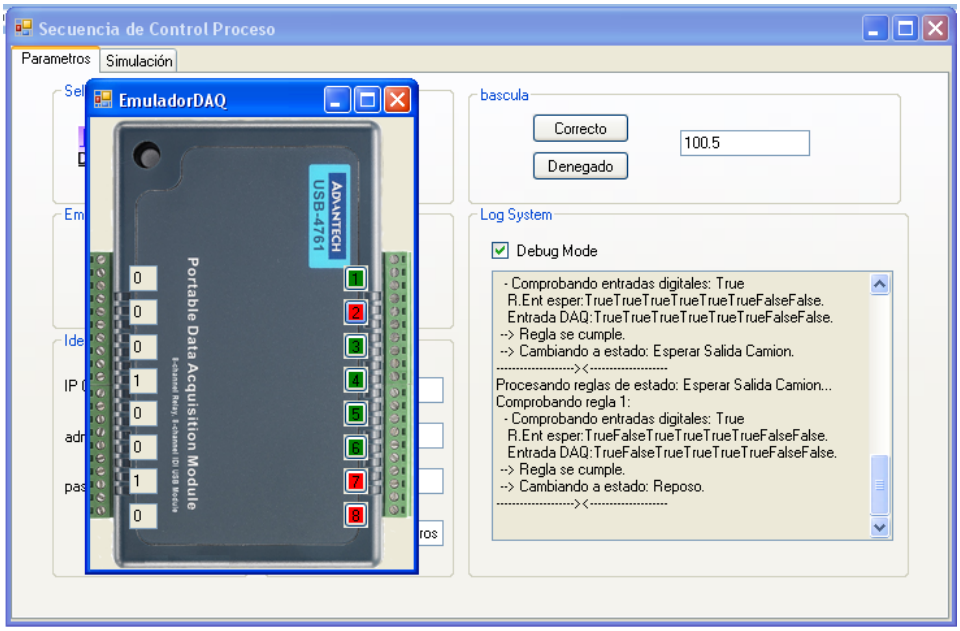

Ilustración 7: simulación paso 4

Ahora se procederá a narrar la maniobra inversa que se ha llevado a cabo. Hasta ahora la simulación ha realizado el procedimiento de entrada del camión a la planta de reciclaje desde el exterior. Ahora el camión que ha realizado la carga o descarga realizará la maniobra de salida de la planta de reciclaje.

Detectando el camión ocurre lo mismo que a su entrada. Automáticamente se lanza el proceso de identificación pero en este caso sólo se hará uso del simulador de RFID.

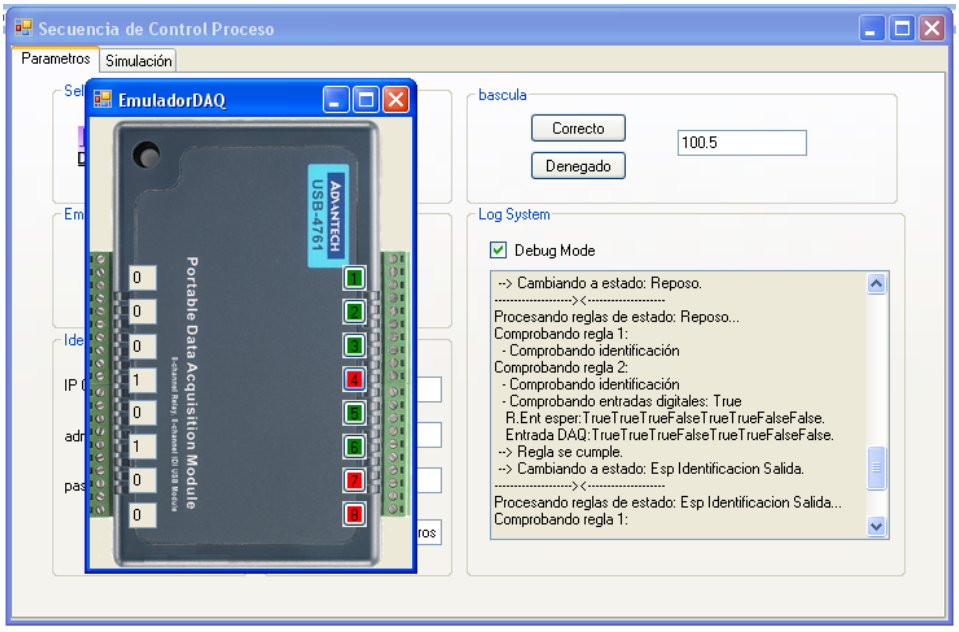

Ilustración 8: Simulación paso 5

El vehículo una vez identificado correctamente se espera que el que se situé en la zona de la báscula.

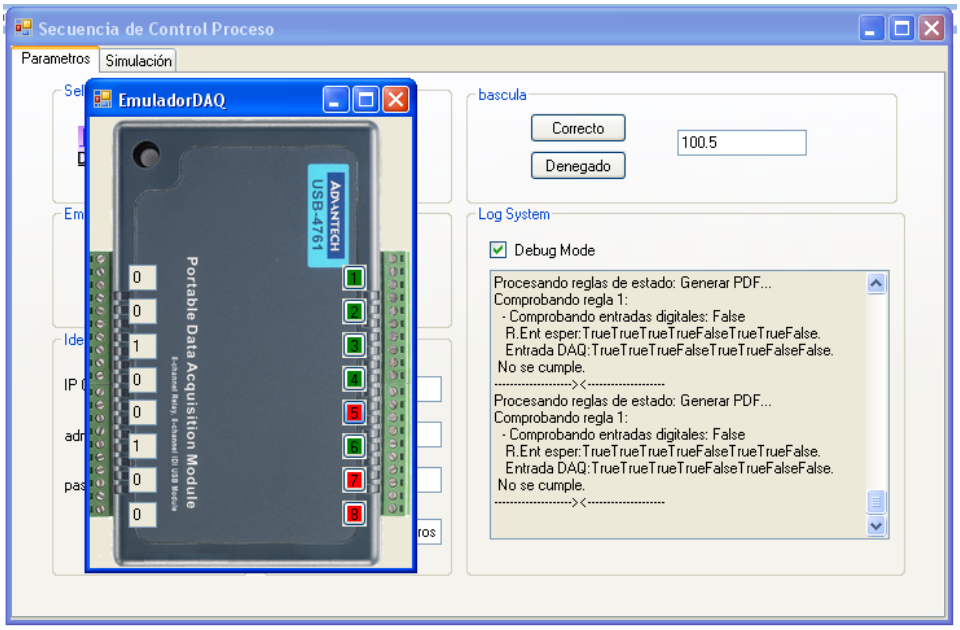

Ilustración 9: simulación paso 6

Ahora el conductor se encuentra en la misma casuística que al entrar. El conductor ha de presionar un botón para que se vuelva a producir la pesada y se genere el ticket.

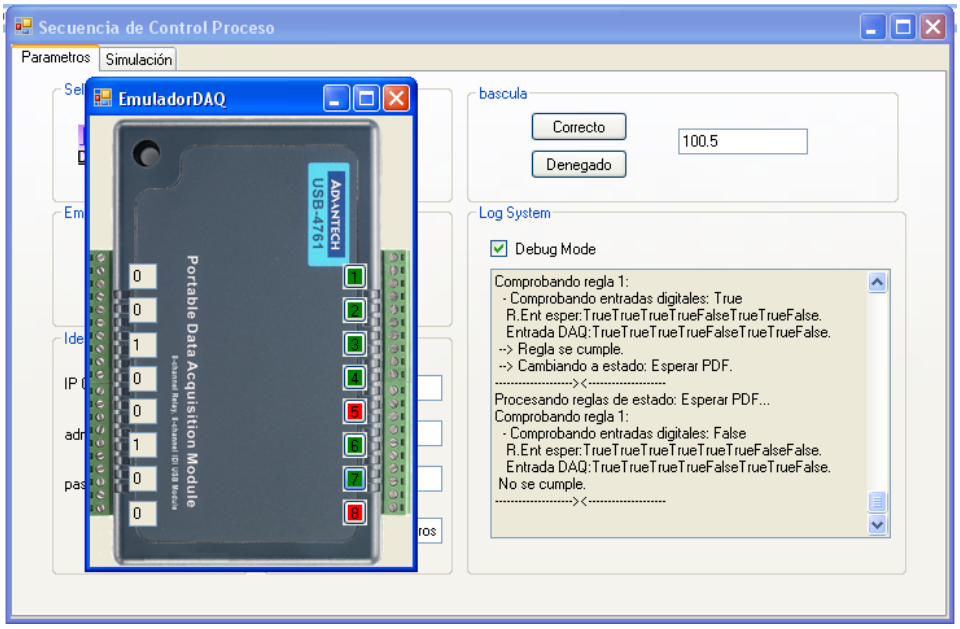

Ilustración 10: simulación paso 7

Ahora el sistema se encuentra que el camión termine la maniobra con éxito y salga de la plataforma accionando el último final de carrera. Una vez realizada esta acción el sistema volverá a estado reposo y concluirá la simulación.

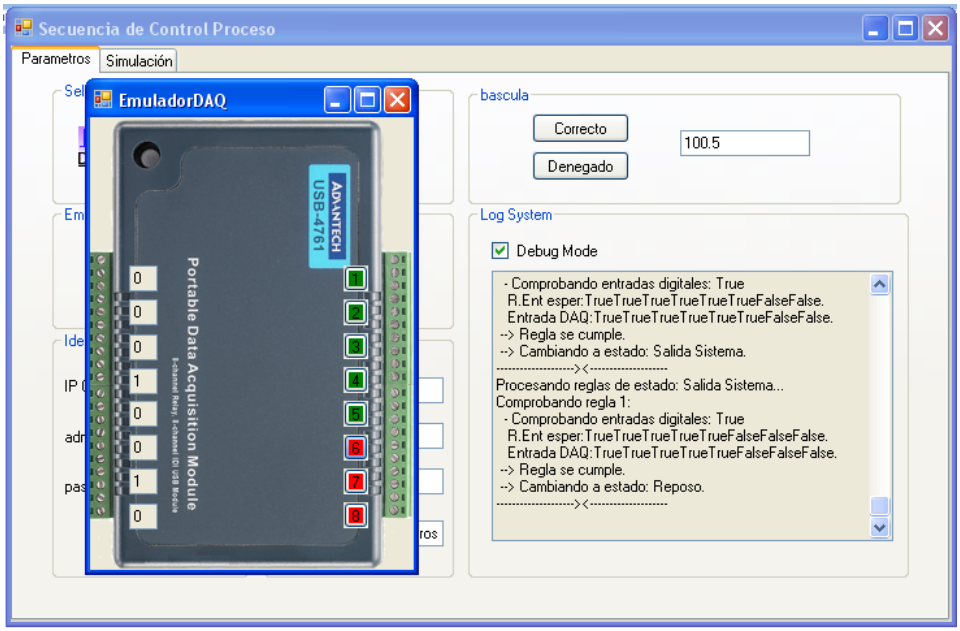

Ilustración 11: simulación paso 8